Infinite Shift-invariant Grouped Multi-task Learning for Gaussian Processes*

Yuyang Wang[1], Roni Khardon[1], and Pavlos Protopapas[2]

[1] Tufts University, Medford, MA USA {ywang02|roni}@cs.tufts.edu
[2] Harvard-Smithsonian Center for Astrophysics, Cambridge, MA USA
pprotopapas@cfa.harvard.edu

Abstract. Multi-task learning leverages shared information among data sets to improve the learning performance of individual tasks. The paper applies this framework for data where each task is a phase-shifted periodic time series. In particular, we develop a novel Bayesian nonparametric model capturing a mixture of Gaussian processes where each task is a sum of a group-specific function and a component capturing individual variation, in addition to each task being phase shifted. We develop an efficient EM algorithm to learn the parameters of the model. As a special case we obtain the Gaussian mixture model and EM algorithm for phased-shifted periodic time series. Furthermore, we extend the proposed model by using a Dirichlet Process prior and thereby leading to an infinite mixture model that is capable of doing automatic model selection. A Variational Bayesian approach is developed for inference in this model. Experiments in regression, classification and class discovery demonstrate the performance of the proposed models using both synthetic data and real-world time series data from astrophysics. Our methods are particularly useful when the time series are sparsely and non-synchronously sampled.

1 Introduction

In many real world problems we are interested in learning multiple tasks while the training set for each task is quite small. For example, in pharmacological studies, we may be attempting to predict the concentration of some drug at different times across multiple patients. Finding a good regression function of an individual patient based only on his or her measurements can be difficult due to insufficient training points for the patient. Instead, by using measurements across all the patients, we may be able to leverage common patterns across patients to obtain better estimates for the population and for each patient individually. Multi-task learning captures this intuition aiming to learn multiple correlated tasks simultaneously. This idea has attracted much interest in the literature and several approaches have been applied to a wide range of domains including medical diagnosis [1], recommendation systems [2] and HIV Therapy Screening [3].

* This is an extended version of our ECML 2010 paper.

Building on the theoretical framework for single-task learning, multi-task learning has recently been formulated by [4] as a multi-task regularization problem in vector-valued Reproducing Kernel Hilbert space.

Several approaches formalizing multi-task learning exist within Bayesian statistics. Considering hierarchical Bayesian models [5, 6], one can view the parameter sharing of the prior among tasks as a form of multi-task learning where evidence from all tasks is used to infer the parameters. Over the past few years, Bayesian models for multi-task learning were formalized using Gaussian processes [7–9]. In this mixed-effect model, information is shared among tasks by having each task combine a common (fixed effect) portion and a task specific portion, each of which is generated by an independent Gaussian process.

Our work builds on this formulation extending it and the associated algorithms in several ways. In particular, we extend the model to include three new aspects. First, we allow the fixed effect to be multi-modal so that each task may draw its fixed effect from a different cluster. Second, we extend the model so that each task may be an arbitrarily phase-shifted image of the original time series. This yields our GMT model: the shift-invariant grouped mixed-effect model. Alternatively, our model can be viewed as a probabilistic extension of the Phased K-means algorithm of [10] that performs clustering for phase-shifted time series data and as a non-parametric Bayesian extension of mixtures of random effects regressions for curve clustering [11]. Finally, unlike the existing models that require the model order to be set a priori, our extension in the DP-GMT model uses a Dirichlet process prior on the mixture proportions so that the number of mixture components is adaptively determined by the data rather than being fixed explicitly.

Our main technical contribution is the inference algorithm for the proposed model. We develop details for the EM algorithm for the GMT model and a Variational EM for DP-GMT optimizing the maximum a posteriori (MAP) estimates for the parameters of the models. Technically, the main insights are in estimating the expectation for the coupled hidden variables (the cluster identities and the task specific portion of the time series) and in solving the regularized least squares problem for a set of phase-shifted observations. In addition, for the DP-GMT, we show that the variational EM algorithm can be implemented with the same complexity as the fixed order GMT without using sampling. Thus the DP-GMT provides an efficient model selection algorithm compared to alternatives such as BIC. As a special case our algorithm yields the (Infinite) Gaussian mixture model for phase shifted time series, which may be of independent interest, and which is a generalization of the algorithms of [10] and [11].

Our model primarily captures regression of time series but because it is a generative model it can be used for class discovery, clustering, and classification. We demonstrate the utility of the model using several experiments with both synthetic data and real-world time series data from astrophysics. The experiments show that our model can yield superior results when compared to the single-task learning and Gaussian mixture models, especially when each individual task is sparsely and non-synchronously sampled. The DP-GMT model

yields results that are competitive with model selection using BIC over the GMT model, at much reduced computational cost.

The remainder of the paper is organized as follows. Section 2 provides an introduction to the multi-task learning problem and its Bayesian interpretation and develops the main assumptions of our model. Section 3 defines the new generative model, Section 4 develops the EM algorithm for it, and the infinite mixture extension is addressed in Section 5. The experimental results are reported in Section 6. Related work is discussed in Section 7 and the final section concludes with a discussion and outlines ideas for future work.

2 Preliminaries

Throughout the paper, scalars are denoted using italics, as in $x, y \in \mathbb{R}$; vectors use bold typeface, as in $\boldsymbol{x}, \boldsymbol{y}$, and x_i denotes the ith entry of $\boldsymbol{x}$. For a vector $\boldsymbol{x}$ and real valued function $f : \mathbb{R} \to \mathbb{R}$, we extend the notation for f to vectors so that $f(\boldsymbol{x}) = [f(x_1), \cdots, f(x_n)]^{\mathrm{T}}$ where the superscript T stands for transposition (and the result is a column vector). $\mathcal{K}(\cdot, \cdot)$ denotes a kernel function associated to some reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ and its norm is denoted as $\|\cdot\|_{\mathcal{H}}$. To keep the notation simple, $\sum_{j=1}^{M}$ is substituted by $\sum_j$ where the index j is not confusing.

2.1 Multi-task learning with kernel

Given training set $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}, i = 1, \cdots, N$, where $\boldsymbol{x}_i = [x_{i1}, x_{i2}, \cdots, x_{id}]^{\mathrm{T}} \in \mathcal{X} \subset \mathbb{R}^d$, single-task learning focuses on finding a function $f : \mathcal{X} \to \mathbb{R}$ that best fits and generalizes the observed data. In the regularization framework, learning f amounts to solving the following variational problem [12, 13]

$$f^* = \operatorname*{argmin}_{f \in \mathcal{H}} \left\{ \sum_i V(f(\boldsymbol{x}_i), y_i) + \lambda \|f\|_{\mathcal{H}}^2 \right\} \tag{1}$$

where $V(\cdot, \cdot)$ is some (typically convex) loss function. The norm $\|\cdot\|_{\mathcal{H}}$ relates to regularity condition on the function where a large norm penalizes non-smooth functions. The regularization parameter λ provides a tradeoff between the loss term and the complexity of the function.

Consider a set of M tasks, with jth task $\mathcal{D}^j = (\boldsymbol{x}_i^j, y_i^j), i = 1, 2, \cdots, n_j$. Multi-task learning seeks to find f^j for each task simultaneously, which, assuming square loss function, can be formulated as the following regularization problem

$$\operatorname*{argmin}_{f^1, \cdots, f^M \in \mathcal{H}} \left\{ \frac{1}{M} \sum_{j=1}^{M} \sum_{i=1}^{n_j} (y_i^j - f^j(\boldsymbol{x}_i^j))^2 + \lambda \mathrm{PEN}(f^1, f^2, \cdots, f^j) \right\} \tag{2}$$

where the penalty term, applying jointly to all the tasks, encodes our prior information on how smooth the functions are, as well as how these tasks are

correlated with each other. For example, setting the penalty term to $\sum_j \|f^j\|_{\mathcal{H}}$ implies that there is no correlation among the tasks. It further decomposes the optimization functional to M separate single-task learning problems. On the other hand, with a shared penalty, the joint regularization can lead to improved performance. Moreover, we can use a norm in RKHS with a *multi-task kernel* to incorporate the penalty term [14]. Formally, consider a vector-valued function $f : \mathcal{X} \to \mathbb{R}^M$ defined as $f \triangleq [f^1, f^2, \cdots, f^M]^{\mathrm{T}}$. Then Equation (2) can be written as

$$\underset{f \in \mathcal{H}}{\operatorname{argmin}} \left\{ \frac{1}{M} \sum_{j=1}^{M} \sum_{i=1}^{n_j} (y_i^j - f^j(\boldsymbol{x}_i^j))^2 + \lambda \|f\|_{\mathcal{H}}^2 \right\} \tag{3}$$

where $\|\cdot\|_{\mathcal{H}}$ is the norm in RKHS with the multi-task kernel $\mathcal{Q} : (\boldsymbol{\Lambda}, \mathcal{X}) \times (\boldsymbol{\Lambda}, \mathcal{X}) \to \mathbb{R}$, where $\boldsymbol{\Lambda} = \{1, 2, \cdots, M\}$. As shown by [4], the *representer theorem* gives the form of the solution to Equation (3)

$$f^{\ell}(\cdot) = \sum_{j=1}^{M} \sum_{i=1}^{n_j} c_i^j \mathcal{Q}((\cdot, \ell), (\boldsymbol{x}_i^j, j)) \tag{4}$$

with norm

$$\|f\|_{\mathcal{Q}}^2 = \sum_{\ell,k} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_k} c_i^\ell c_j^k \mathcal{Q}((\boldsymbol{x}_i^\ell, \ell), (\boldsymbol{x}_j^k, k)).$$

Let $\mathbf{C} = [c_1^1, c_2^1, \cdots, c_{n_M}^M]^{\mathrm{T}}, \mathbf{Y} = [y_1^1, y_2^1, \cdots, y_{n_M}^M]^{\mathrm{T}} \in \mathbb{R}^{\sum_j n_j}$ and $\mathbf{X} = [\boldsymbol{x}_1^1, \boldsymbol{x}_2^1, \cdots, \boldsymbol{x}_{n_M}^M]$, then the coefficients $\{c_i^j\}$ are given by the following linear system

$$(\mathbf{Q} + \lambda \mathbb{I})\mathbf{C} = \mathbf{Y} \tag{5}$$

where $\mathbf{Q} \in \mathbb{R}^{\sum_j n_j \times \sum_j n_j}$ is the kernel matrix formed by $\mathbf{X}$.

2.2 Bayesian formulation

A Gaussian process is a functional extension for Multivariate Gaussian distributions. In the Bayesian literature, it has been widely used in statistical models by substituting a parametric latent function with a stochastic process with a Gaussian prior [15]. More precisely, under the single-task setting a simple Gaussian regression model is given by

$$y = f(\boldsymbol{x}) + \epsilon$$

where f's prior is a zero mean Gaussian process with covariance function $\mathcal{K}$ and ϵ is independent zero mean white noise with variance σ^2. Given data set $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}, i = 1, \cdots, N$, let $\mathbf{K} = (\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j))_{i,j}$, then $\mathbf{f} \triangleq [f(\boldsymbol{x}_1), \cdots, f(\boldsymbol{x}_N)]^{\mathrm{T}} \sim \mathcal{N}(\mathbf{0}, \mathbf{K})$ and the posterior on $\mathbf{f}$ is given by

$$\Pr(\mathbf{f}|\mathcal{D}) = \mathcal{N}(\mathbf{K}(\sigma^2\mathbb{I} + \mathbf{K})^{-1}\boldsymbol{y}, \sigma^2(\sigma^2\mathbb{I} + \mathbf{K})^{-1}\mathbf{K}).$$

The predictive distribution for some test point $\boldsymbol{x}_*$ distinct from the training examples is

$$\begin{aligned}\Pr(f(\boldsymbol{x}_*)|\boldsymbol{x}_*, \mathcal{D}) &= \int \Pr(f(\boldsymbol{x}_*)|\boldsymbol{x}_*, f) \Pr(f|\mathcal{D}) df \\ &= \mathcal{N}\left(\mathbf{k}(\boldsymbol{x}_*)^{\mathrm{T}}(\sigma^2\mathbb{I} + \mathbf{K})^{-1}\boldsymbol{y}, \mathcal{K}(\boldsymbol{x}_*, \boldsymbol{x}_*) - \mathbf{k}(\boldsymbol{x}_*)^{\mathrm{T}}(\sigma^2\mathbb{I} + \mathbf{K})^{-1}\mathbf{k}(\boldsymbol{x}_*)\right)\end{aligned}$$

where $\mathbf{k}(\boldsymbol{x}_*) = [\mathcal{K}(\boldsymbol{x}_1, \boldsymbol{x}_*), \cdots, \mathcal{K}(\boldsymbol{x}_N, \boldsymbol{x}_*)]^{\mathrm{T}}$. Furthermore, under square loss function, the optimizer of Equation (1) is equal to the expectation of the predictive distribution [15]. Finally, a Gaussian process f corresponds to a RKHS $\mathcal{H}$ with kernel $\mathcal{K}$ such that

$$\mathrm{cov}[f(\boldsymbol{x}), f(\boldsymbol{y})] = \mathcal{K}(\boldsymbol{x}, \boldsymbol{y}) \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}. \tag{6}$$

In this way, we can express a prior on functions f using a zero mean Gaussian process [16][3]

$$f \sim \exp\left\{-\frac{1}{2}\|f\|_{\mathcal{H}}^2\right\}. \tag{7}$$

Applying this framework in the context of multi-task learning, the model is given by

$$\boldsymbol{y}_i^j = f^j(\boldsymbol{x}_i^j) + \epsilon_{ij}$$

where f^j are zero mean Gaussian processes and ϵ_{ij} captures i.i.d. zero-mean noise with variance σ^2. [9] formalize the connection between *multi-task kernel* $\mathcal{Q}$ and covariance function among $\{f^j\}$ using

$$\mathrm{cov}[f^i(\boldsymbol{x}), f^j(\boldsymbol{x}')] = \mathcal{Q}((\boldsymbol{x}, i), (\boldsymbol{x}', j)), \quad i, j = 1, \cdots, M. \tag{8}$$

2.3 Basic Model Assumptions

Given data $\{\mathcal{D}^j\}$, the so-called nonparametric Bayesian mixed-effect model [16, 9] captures each task f^j with respect to $\mathcal{D}^j$ using a sum of an average effect function and an individual variation for each specific task,

$$f^j(x) = \bar{f}(x) + \tilde{f}^j(x), \quad j = 1, \cdots, M.$$

This assumes that the fixed-effect (mean function) $\bar{f}$ is sufficient to capture the behavior of the data, an assumption that is problematic for distributions with several modes. To address this, we introduce a mixture model allowing for multiple modes (just like standard Gaussian mixture model (GMM)), but maintaining the formulation using Gaussian processes. This amounts to adding a group effect structure and leads to the following assumption:

[3] In general, a Gaussian process can not be thought of as a distribution on the RKHS, because with probability 1, one can find a Gaussian process such that its sample path does not belong to the RKHS. However, the equivalence holds between the RKHS and the expectation of a Gaussian process conditioned on a finite number of observations. For more details on the relationship between RKHS and Gaussian processes we refer interested reader to [17].

Assumption 1 *For each j and $\boldsymbol{x} \in \mathcal{X}$,*

$$f^j(\boldsymbol{x}) = \bar{f}_{z_j}(\boldsymbol{x}) + \tilde{f}^j(\boldsymbol{x}), \quad j = 1, \cdots, M \tag{9}$$

where $\{\bar{f}_s\}, s = 1, \cdots, k$ and $\tilde{f}^j$ are zero-mean Gaussian processes and $z_j \in \{1, \cdots, k\}$. In addition, $\{\bar{f}_s\}$ and $\tilde{f}^j$ are assumed to be mutually independent.

With the grouped-effect model and groups predefined, one can define a kernel that relates (with non zero similarity) only points from the same example or points for different examples but the same center as follows

$$\mathcal{Q}((\boldsymbol{x}, i), (\boldsymbol{x}', j)) = \delta_{z_i, z_j} \overline{\mathcal{K}}_{z_i}(\boldsymbol{x}, \boldsymbol{x}') + \delta_{i,j} \widetilde{\mathcal{K}}_i(\boldsymbol{x}, \boldsymbol{x}')$$

where

$$\begin{cases} \overline{\mathcal{K}}_{z_i}(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{cov}[\bar{f}_{z_i}(\boldsymbol{x}), \bar{f}_{z_i}(\boldsymbol{x}')], \\ \widetilde{\mathcal{K}}_i(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{cov}[\tilde{f}^j(\boldsymbol{x}), \tilde{f}^j(\boldsymbol{x}')]. \end{cases}$$

However, in our work the groups are not known in advance and we cannot use this formulation. Instead we use a single kernel to relate all tasks.

The second extension allows us to handle phase shifted time series. In some applications, we face the challenge of learning a periodic function $f^j : \mathbb{R} \to \mathbb{R}$ on a single period T from samples $\mathcal{D} = \{\boldsymbol{x}^j, \boldsymbol{y}^j\}, \boldsymbol{x}^j, \boldsymbol{y}^j \in \mathbb{R}^{n_j}, j = 1, \cdots, M$, where similar functions in a group differ only in their phase. In the following assumption, the model of primary focus in this paper is presented, which extends the mixed-effect model to capture both shift-invariance and the clustering property.

Assumption 2 *For each j and $x \in [0, T)$,*

$$f^j(x) = [\bar{f}_{z_j} * \delta_{t_j}](x) + \tilde{f}^j(x), \quad , j = 1, \cdots, M \tag{10}$$

where $z_j \in \{1, \cdots, k\}$, $\{\bar{f}_s\}, s = 1, \cdots, k$ and $\tilde{f}^j$ are zero-mean Gaussian processes, $*$ stands for circular convolution and δ_{t_j} is the Dirac δ function with support at $t_j \in [0, T)$.[4] In addition, $\{\bar{f}_s\}, \tilde{f}^j$ are assumed to be mutually independent.

[4] Given a periodic function f with period T, its circular convolution with another function h is defined as

$$(f * h)(t) \triangleq \int_{t_0}^{t_0+T} f(t-\tau) h(\tau) d\tau$$

where t_0 is arbitrary in $\mathbb{R}$ and $f * h$ is also a periodic function with period T. Using the definition we see that,

$$f * \delta_{t_j}(t) = f(t - t_j),$$

and thus $*$ performs a right shift of f or in other words performs a phase shift of t_j on f.

3 Shift-invariant Grouped mixed-effect model

In Assumption 1, if we know the cluster assignment of each task, then the model decomposes to k mixed-effect models which is the case investigated in [9, 4]. Similar results can be obtained for Assumption 2. However, prior knowledge of cluster membership is often not realistic. In this section, based on Assumption 2, a probabilistic generative model is formulated to capture the case of unknown clusters. We start by formally defining the generative model, which we call *Shift-invariant Grouped mixed-effect Model* (GMT). In this model, k group effect functions are assumed to share the same Gaussian prior characterized by $\mathcal{K}_0$. The individual effect functions are Gaussian processes with covariance function $\mathcal{K}$. The model is shown in Figure 1 and it is characterized by parameter set $\mathcal{M} = \{\mathcal{K}_0, \mathcal{K}, \boldsymbol{\alpha}, \{t_j\}, \sigma^2\}$ and summarized as follows

1. Draw $\bar{f}_s|\mathcal{K}_0 \sim \exp\left\{-\frac{1}{2}\|\bar{f}_s\|^2_{\mathcal{H}_0}\right\}, \quad s = 1, 2, \cdots, k$
2. For the jth time series
   - Draw $z_j|\boldsymbol{\alpha} \sim \text{Discrete}(\boldsymbol{\alpha})$
   - Draw $\tilde{f}^j|\mathcal{K} \sim \exp\left\{-\frac{1}{2}\|\tilde{f}^j\|^2_{\mathcal{H}}\right\}$
   - Draw $\boldsymbol{y}^j|z_j, f^j, \boldsymbol{x}^j, t_j, \sigma^2 \sim \mathcal{N}\left(f^j(\boldsymbol{x}^j), \sigma^2\mathbb{I}\right)$, where $f^j = \bar{f}_{z_j} * \delta_{t_j} + \tilde{f}^j$

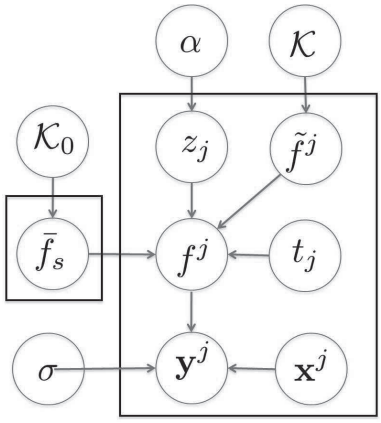

Fig. 1. GMT: Plate graph

where $\boldsymbol{\alpha}$ is the mixture proportion. Additionally, denote $\mathcal{X} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \cdots, \boldsymbol{x}^M\}$ and $\mathcal{Y} = \{\boldsymbol{y}^1, \boldsymbol{y}^2, \cdots, \boldsymbol{y}^M\}$, where $\boldsymbol{x}^j$ are the time points when each time series is sampled and $\boldsymbol{y}^j$ are the corresponding observations.

We assume that the group effect kernel $\mathcal{K}_0$ and the number of centers k are known. The assumption on $\mathcal{K}_0$ is reasonable, in that normally we can get more information on the shape of the mean waveforms, thereby making it possible to design kernel for $\mathcal{H}_0$. On the other hand, the individual variations are more

arbitrary and therefore $\mathcal{K}$ is not assumed to be known. The assumption that k is known requires some form of model selection. An extension using a non-parametric Bayesian model, the *Dirichlet process* [18], that does not limit k is discussed in the section 5. The group effect $\{\bar{f}_s\}$, individual shifts $\{t_j\}$, noise variance σ^2 and the kernel for individual variations $\mathcal{K}$ are unknown and need to be estimated. The cluster assignments $\{z_j\}$ and individual variation $\{\tilde{f}^j\}$ are treated as hidden variables. Note that one could treat $\{\bar{f}_s\}$ too as hidden variables, but we prefer to get a concrete estimate for these variables because of their role as the mean waveforms in our model.

The model above is a standard model for regression. We propose to use it for classification by learning a mixture model for each class and using the *Maximum A Posteriori* (MAP) probability for the class for classification. In particular, consider a training set that has L classes, where the jth instance is given by $\mathcal{D}^j = (\boldsymbol{x}^j, \boldsymbol{y}^j, o^j) \in \mathbb{R}^{n_j} \times \mathbb{R}^{n_j} \times \{1, 2, \cdots, L\}$. Each observation $(\boldsymbol{x}^j, \boldsymbol{y}^j)$ is given a label from $\{1, 2, \cdots, L\}$. The problem is to learn the model M_ℓ for each class (L in total) separately and the classification rule for a new instance $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ is given by

$$o = \underset{\ell=\{1,\cdots,L\}}{\operatorname{argmax}} \Pr(\boldsymbol{y}^*|\boldsymbol{x}^*; \mathrm{M}_\ell) \Pr(\ell). \tag{11}$$

As we show in our experiments, the generative model can provide explanatory power for the application while giving excellent classification performance.

4 Parameter Estimation

Given data set $\mathcal{D} = \{\boldsymbol{x}^j, \boldsymbol{y}^j\} = \{x_i^j, y_i^j\}, i = 1, \cdots, n_j, j = 1, \cdots, M$, the learning process aims to find the MAP estimates of the parameter set $\mathcal{M} = \{\boldsymbol{\alpha}, \{\bar{f}_s\}, \{t_j\}, \sigma^2, \mathcal{K}\}$

$$\mathcal{M}^* = \underset{\mathcal{M}}{\operatorname{argmax}} \left(\Pr(\mathcal{Y}|\mathcal{X}; \mathcal{M}) \times \Pr[\{\bar{f}_s\}; \mathcal{K}_0] \right). \tag{12}$$

The direct optimization of Equation (12) is analytically intractable because of coupled sums that come from the mixture distribution. To solve this problem, we resort to the EM algorithm [19]. The EM algorithm is an iterative method for optimizing the maximum likelihood (ML) or MAP estimates of the parameters in the context of hidden variables. In our case, the hidden variables are $\mathbf{z} = \{z_j\}$ (which is the same as in standard GMM), and $\mathbf{f} = \{\mathbf{f}_j \triangleq \tilde{f}^j(\boldsymbol{x}^j)\}, j = 1, \cdots, M$. The algorithm iterates between the following expectation and maximization steps until it converges to a local maximum.

4.1 Expectation step

In the **E**-step, we calculate

$$Q(\mathcal{M}, \mathcal{M}^g) = \mathbb{E}_{\{\mathbf{z},\mathbf{f}|\mathcal{X},\mathcal{Y};\mathcal{M}^g\}} \left[\log \left\{ \Pr(\mathcal{Y}, \mathbf{f}, \mathbf{z}|\mathcal{X}; \mathcal{M}) \times \Pr[\{\bar{f}_s\}; \mathcal{K}_0] \right\} \right] \tag{13}$$

where $\mathcal{M}^g$ stands for estimated parameters from the last iteration. For our model, the difficulty comes from estimating the expectation with respect to the coupled latent variables $\{\mathbf{z}, \mathbf{f}\}$. In the following, we show how this can be done. First notice that,

$$\Pr(\mathbf{z}, \mathbf{f}|\mathcal{X}, \mathcal{Y}; \mathcal{M}^g) = \prod_j \Pr(z_j, \mathbf{f}_j|\mathcal{X}, \mathcal{Y}; \mathcal{M}^g)$$

and further that

$$\Pr(z_j, \mathbf{f}_j|\mathcal{X}, \mathcal{Y}; \mathcal{M}^g) = \Pr(z_j|\boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g) \times \Pr(\mathbf{f}_j|z_j, \boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g). \tag{14}$$

The first term in Equation (14) can be further written as

$$\begin{aligned}\Pr(z_j|\boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g) &\propto \Pr(z_j; \mathcal{M}^g) \Pr(\boldsymbol{y}^j|z_j, \boldsymbol{x}^j; \mathcal{M}^g) \\ &= \Pr(z_j; \mathcal{M}^g) \int \Pr(\boldsymbol{y}^j, \mathbf{f}_j|z_j, \boldsymbol{x}^j; \mathcal{M}^g) d\mathbf{f}_j \\ &= \Pr(z_j; \mathcal{M}^g) \int \Pr(\boldsymbol{y}^j|\mathbf{f}_j, z_j, \boldsymbol{x}^j; \mathcal{M}^g) \Pr(\mathbf{f}_j; \mathcal{M}^g) d\mathbf{f}_j\end{aligned} \tag{15}$$

where $\Pr(z_j; \mathcal{M}^g)$ is specified by the parameters estimated from last iteration. Since z_j is given, the second term is the marginal distribution that can be calculated using a Gaussian process regression model. In particular, denoting $\bar{\mathbf{f}}^j = \bar{f}_{z_j} * \delta_{t_j}(\boldsymbol{x}^j)$ we get

$$\begin{bmatrix}\boldsymbol{y}^j \\ \mathbf{f}^j\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}\bar{\mathbf{f}}^j \\ 0\end{bmatrix}, \begin{bmatrix}\mathbf{K}_j^g + \sigma^2\mathbb{I} & \mathbf{K}_j^g \\ \mathbf{K}_j^g & \mathbf{K}_j^g\end{bmatrix}\right)$$

where $\mathbf{K}_j^g$ is the kernel matrix for the jth task using parameters from last iteration, i.e. $\mathbf{K}_j^g = (\mathcal{K}(x_i^j, x_l^j))_{il}$. Therefore, the marginal distribution is

$$\boldsymbol{y}^j|z_j \sim \mathcal{N}(\bar{\mathbf{f}}^j, \mathbf{K}_j^g + \sigma^2\mathbb{I}). \tag{16}$$

Next consider the second term in Equation (14). Given z_j, we know that $f^j = \bar{f}_{z_j} + \tilde{f}^j$, i.e. there is no uncertainty about the identity of $\bar{f}_{z_j}$ and therefore the calculation amounts to estimating the posterior distribution under standard Gaussian process regression, that is

$$\begin{aligned}\boldsymbol{y}^j - \bar{\mathbf{f}}^j &\sim \mathcal{N}(\tilde{f}^j(\boldsymbol{x}^j), \sigma^2\mathbb{I}) \\ \tilde{f}^j &\sim \exp\left\{-\frac{1}{2}\|\tilde{f}^j\|_{\mathcal{K}}^2\right\}\end{aligned}$$

and the conditional distribution is given by

$$\mathbf{f}_j|z_j, \boldsymbol{x}^j, \boldsymbol{y}^j \sim \mathcal{N}(\mu_j^g, \mathbf{C}_j^g) \tag{17}$$

where μ_j^g is the posterior mean

$$\mu_j^g = \mathbf{K}_j^g(\mathbf{K}_j^g + \sigma^2\mathbb{I})^{-1}(\boldsymbol{y}^j - \bar{\mathbf{f}}^j) \tag{18}$$

and $\mathbf{C}_j^g$ is the posterior covariance of $\mathbf{f}_j$

$$\mathbf{C}_j^g = \mathbf{K}_j^g - \mathbf{K}_j^g(\mathbf{K}_j^g + \sigma^2\mathbb{I})^{-1}\mathbf{K}_j^g. \tag{19}$$

Since Equation (15) is multinomial and $\mathbf{f}_j$ is Normal in (17), the marginal distribution of $\mathbf{f}_j$ is a Gaussian mixture distribution given by

$$\begin{aligned}\Pr(\mathbf{f}_j|\boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g) = \sum_s &\Pr(z_j = s|\boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g) \\ &\times \mathcal{N}\left(\mu_j, \mathbf{C}_j|z_j = s; \mathcal{M}^g\right), \quad s = 1, \cdots, k.\end{aligned}$$

To work out the concrete form of $Q(\mathcal{M}, \mathcal{M}^g)$, denote $z_{il} = 1$ iff $z_i = l$. Then the complete data likelihood can be reformulated as

$$\begin{aligned}\mathcal{L} &= \Pr(\mathcal{Y}, \mathbf{f}, \mathbf{z}; \mathcal{X}, \mathcal{M}) \\ &= \prod_j \prod_s \left[\alpha_s \Pr(\boldsymbol{y}^j, \mathbf{f}_j|z_j = s; \mathcal{M})\right]^{z_{js}} \\ &= \prod_j \prod_s \left[\alpha_s \Pr(\boldsymbol{y}^j|\mathbf{f}_j, z_j = s; \mathcal{M}) \Pr(\mathbf{f}_j; \mathcal{M})\right]^{z_{js}}\end{aligned}$$

where we have used the fact that exactly one z_{js} is 1 for each j and included the last term inside the product over s for convenience. Then Equation (13) can be written as

$$Q(\mathcal{M}, \mathcal{M}^g) = -\frac{1}{2}\sum_s \|f_s\|^2_{\mathcal{H}_0} + \mathbb{E}_{\{\mathbf{z},\mathbf{f}|\mathcal{X},\mathcal{Y};\mathcal{M}^g\}}\left[\log \mathcal{L}\right].$$

Denote the second term by $\widetilde{Q}$. By a version of Fubini's theorem [20] we have

$$\begin{aligned}\widetilde{Q} &= \mathbb{E}_{\{\mathbf{z}|\mathcal{X},\mathcal{Y};\mathcal{M}^g\}}\mathbb{E}_{\{\mathbf{f}|\mathbf{z},\mathcal{X},\mathcal{Y};\mathcal{M}^g\}}\left[\log \mathcal{L}\right] \\ &= \sum_{\mathbf{z}} \Pr(\mathbf{z}|\mathcal{X}, \mathcal{Y}; \mathcal{M}^g)\Bigg\{\sum_j \sum_s z_{js} \\ &\qquad \times \int d\Pr(\mathbf{f}_j|z_j = s) \log\left[\alpha_s \Pr(\boldsymbol{y}^j|\mathbf{f}_j, z_j = s; \mathcal{M}) \Pr(\mathbf{f}_j; \mathcal{M})\right]\Bigg\}.\end{aligned} \tag{20}$$

Now because the last term in Equation (20) does not include any z_i, the equation can be further decomposed as

$$\begin{aligned}\widetilde{Q} &= \sum_j \sum_s \left(\sum_{\mathbf{z}} \Pr(\mathbf{z}|\mathcal{X}, \mathcal{Y}; \mathcal{M}^g) z_{js}\right) \\ &\qquad \times \left\{\int d\Pr(\mathbf{f}_j|z_j = s) \log[\alpha_s \Pr(\boldsymbol{y}^j|\mathbf{f}_j, z_j = s; \mathcal{M}) \Pr(\mathbf{f}_j; \mathcal{M})]\right\} \\ &= \sum_j \sum_s \gamma_{js} \int d\Pr(\mathbf{f}_j|z_j = s) \log\left[\alpha_s \Pr(\boldsymbol{y}^j|\mathbf{f}_j, z_j = s; \mathcal{M}) \Pr(\mathbf{f}_j; \mathcal{M})\right] \\ &= \sum_j \sum_s \gamma_{js} \mathbb{E}_{\{\mathbf{f}_j|z_j=s,\boldsymbol{x}^j,\boldsymbol{y}^j;\mathcal{M}^g\}}\left[\log \alpha_s + \log\left(\Pr(\boldsymbol{y}^j|\mathbf{f}_j, z_j = s; \mathcal{M})\right) + \log\left(\Pr(\mathbf{f}_j; \mathcal{M})\right)\right]\end{aligned} \tag{21}$$

where

$$\gamma_{js} = \mathbb{E}[z_{js}|\boldsymbol{y}^j, \boldsymbol{x}^j; \mathcal{M}^g] = \frac{\Pr(z_j = s|\boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g)}{\sum_s \Pr(z_j = s|\boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g)} \tag{22}$$

can be calculated from Equation (15) and (16) and γ_{js} can be viewed as a fractional label indicating how likely the jth task is to belong to the sth group. Recall that $\Pr(\boldsymbol{y}^j|\mathbf{f}_j, z_j = s)$ is a normal distribution given by $\mathcal{N}([\bar{f}_{z_j} * \delta_{t_j}](\boldsymbol{x}^j) + \mathbf{f}_j, \sigma^2\mathbb{I})$ and $\Pr(\mathbf{f}_j; \mathcal{M})$ is a standard multivariate Gaussian distribution determined by its prior

$$\Pr(\mathbf{f}_j; \mathcal{M}) = \frac{1}{\sqrt{(2\pi)^{n_j}|\mathbf{K}_j|}} \exp\left\{-\frac{1}{2}\mathbf{f}_j^T \mathbf{K}_j^{-1} \mathbf{f}_j\right\}.$$

Using these facts and Equation (21), $Q(\mathcal{M}, \mathcal{M}^g)$ can be re-formulated as

$$\begin{aligned} Q(\mathcal{M}, \mathcal{M}^g) = &-\frac{1}{2}\sum_s \|\bar{f}_s\|^2_{\mathcal{H}_0} - \sum_j n_j \log\sigma + \sum_j \sum_s \gamma_{js} \log\alpha_s \\ &- \frac{1}{2\sigma^2}\sum_j \sum_s \gamma_{js} \mathbb{E}_{\{\mathbf{f}_j|z_j=s, \boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g\}}\left[\|\boldsymbol{y}^j - [\bar{f}_s * \delta_{t_j}](\boldsymbol{x}^j) - \mathbf{f}_j\|^2\right] \\ &- \frac{1}{2}\sum_j \log|\mathbf{K}_j| - \frac{1}{2}\sum_j \sum_s \gamma_{js} \mathbb{E}_{\{\mathbf{f}_j|z_j=s, \boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g\}}\left(\mathbf{f}_j^{\mathrm{T}} \mathbf{K}_j^{-1} \mathbf{f}_j\right) \end{aligned} \tag{23}$$

We next develop explicit closed forms for the remaining expectations. For the first, note that for $\boldsymbol{x} \sim \mathcal{N}(\mu, \boldsymbol{\Sigma})$ and a constant vector $\mathbf{a}$,

$$\begin{aligned} \mathbb{E}[\|\mathbf{a} - \boldsymbol{x}\|^2] &= \mathbb{E}[\|\mathbf{a}\|^2 - 2\langle \mathbf{a}, \boldsymbol{x}\rangle + \|\boldsymbol{x}\|^2] \\ &= \|\mathbf{a}\|^2 - 2\langle \mathbf{a}, \mathbb{E}[\boldsymbol{x}]\rangle + \mathbb{E}[\boldsymbol{x}]^2 + \mathrm{Tr}(\boldsymbol{\Sigma}) \\ &= \|\mathbf{a} - \mu\|^2 + \mathrm{Tr}(\boldsymbol{\Sigma}). \end{aligned}$$

Therefore the expectation is

$$\begin{aligned} \mathbb{E}_{\{\mathbf{f}_j|z_j=s, \boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g\}}\left[\|\boldsymbol{y}^j - [\bar{f}_s * \delta_{t_j}](\boldsymbol{x}^j) - \mathbf{f}_j\|^2\right] &= \frac{1}{2\sigma^2}\sum_j \mathrm{Tr}(\mathbf{C}_j^g) \\ &+ \frac{1}{2\sigma^2}\sum_j \sum_s \gamma_{js}\left(\|\boldsymbol{y}^j - [f_s * \delta_{t_j}](\boldsymbol{x}^j) - \mu_{js}\|^2\right) \end{aligned} \tag{24}$$

where $\mu_{js} = \mathbb{E}_{\{\mathbf{f}_j|z_j=s, \boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g\}}[\mathbf{f}_j]$ is as in Equation (18) where we set $z_j = s$ explicitly.

For the second expectation we have

$$\begin{aligned} \mathbb{E}_{\{\mathbf{f}_j|z_j=s, \boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g\}}\left(\mathbf{f}_j^{\mathrm{T}} \mathbf{K}_j^{-1} \mathbf{f}_j\right) &= \mathbb{E}_{\{\mathbf{f}_j|z_j=s, \boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g\}}\left[\mathrm{Tr}\left(\mathbf{f}_j^{\mathrm{T}} \mathbf{K}_j^{-1} \mathbf{f}_j\right)\right] \\ &= \mathbb{E}_{\{\mathbf{f}_j|z_j=s, \boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g\}}\left[\mathrm{Tr}\left(\mathbf{K}_j^{-1} \mathbf{f}_j \mathbf{f}_j^{\mathrm{T}}\right)\right] \\ &= \mathrm{Tr}\left(\mathbb{E}_{\{\mathbf{f}_j|z_j=s, \boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M}^g\}}[\mathbf{K}_j^{-1} \mathbf{f}_j \mathbf{f}_j^{\mathrm{T}}]\right) \\ &= \mathrm{Tr}\left(\mathbf{K}_j^{-1}(\mathbf{C}_j^g + \mu_{js}^g(\mu_{js}^g)^{\mathrm{T}})\right). \end{aligned}$$

4.2 M-step

In this step, we aim to find

$$\mathcal{M}^* = \underset{\mathcal{M}}{\operatorname{argmax}}\, Q(\mathcal{M}, \mathcal{M}^g) \tag{25}$$

and use $\mathcal{M}^*$ to update the model parameters. Using the results above this can be decomposed into three separate optimization problems as follows:

$$\begin{aligned}\mathcal{M}^* = \underset{\mathcal{M}}{\operatorname{argmax}}\; & Q_1((\{\bar{f}_s\}, \{\delta_{t_j}\}, \sigma)) \\ & + Q_2(\mathcal{K}) + \left\{ \sum_j \sum_s \gamma_{js} \log \alpha_s \right\}.\end{aligned}$$

That is, $\boldsymbol{\alpha}$ can be estimated easily using its separate term, Q_1 is only a function of $(\{f_s\}, \{t_j\}, \sigma)$ and Q_2 depends only on $\mathcal{K}$, and we have

$$\begin{aligned}Q_1(\{\bar{f}_s\}, \{t_j\}, \sigma^2) = & \frac{1}{2} \sum_s \|\bar{f}_s\|^2_{\mathcal{K}_0} + \sum_j n_j \log \sigma + \frac{1}{2\sigma^2} \sum_j \mathrm{Tr}(\mathbf{C}_j^g) \\ & + \frac{1}{2\sigma^2} \sum_j \sum_s \gamma_{js} \left(\|\boldsymbol{y}^j - [f_s * \delta_{t_j}](\boldsymbol{x}^j) - \mu_{js}\|^2 \right)\end{aligned} \tag{26}$$

and

$$Q_2(\mathcal{K}) = -\frac{1}{2} \sum_j \log |\mathbf{K}_j| - \frac{1}{2} \sum_j \sum_s \gamma_{js} \mathrm{Tr} \left(\mathbf{K}_j^{-1} (\mathbf{C}_j^g + \mu_{js}^g (\mu_{js}^g)^{\mathrm{T}}) \right). \tag{27}$$

The optimizations for Q_1 and Q_2 are described separately in the following two subsections.

Learning $\{\bar{f}_s\}, \{t_j\}, \sigma^2$ To optimize Equation (26) we assume first that σ is given. In this case, optimizing $\{\bar{f}_s\}, \{t_j\}$ decouples into k sub-problems, finding sth group effect $\bar{f}_s$ and its corresponding shift $\{t_j\}$. Denoting the residual $\tilde{\boldsymbol{y}}^j = \boldsymbol{y}^j - \mu_{js}$, where $\mu_{js} = \mathbb{E}[\mathbf{f}_j | \boldsymbol{y}^j, z_j = s]$, the problem becomes

$$\underset{f \in \mathcal{H}_0, t_1, \cdots, t_M \in [0,T)}{\operatorname{argmin}} \left\{ \frac{1}{2\sigma^2} \sum_j \gamma_{js} \sum_{i=1}^{n_j} (\tilde{\boldsymbol{y}}_i^j - [f * \delta_{t_j}](\boldsymbol{x}_i^j))^2 + \frac{1}{2} \|f\|^2_{\mathcal{H}_0} \right\}. \tag{28}$$

Note that different $\boldsymbol{x}^j, \boldsymbol{y}^j$ have different dimensions n_j and they are not assumed to be sampled at regular intervals. For further development, following [9], it is useful to introduce the distinct vector $\breve{\boldsymbol{x}} \in \mathbb{R}^{\mathbb{N}}$ whose component are the distinct elements of $\mathcal{X}$. For example if $\boldsymbol{x}^1 = [1, 2, 3]^T, \boldsymbol{x}^2 = [2, 3, 4, 5]^T$, then $\breve{\boldsymbol{x}} = [1, 2, 3, 4, 5]^T$. For jth task, let the binary matrix C^k be such that

$$\boldsymbol{x}^j = C^j \cdot \breve{\boldsymbol{x}}, \quad f(\boldsymbol{x}^j) = C^j \cdot f(\breve{\boldsymbol{x}}).$$

That is, C^j extracts the values corresponding to the jth task from the full vector. If $\{t_j\}$ are fixed, then the optimization in Equation (28) is standard and the representer theorem gives the form of the solution as

$$f(\cdot) = \sum_{i=1}^{\mathbb{N}} c_i \mathcal{K}_0(\breve{x}_i, \cdot). \tag{29}$$

Denoting the kernel matrix as $\mathfrak{K} = \mathcal{K}_0(\breve{x}_i, \breve{x}_j), i, j = 1, \cdots, \mathbb{N}$, $\mathbf{c} = [c_1, \cdots, c_{\mathbb{N}}]^T$ and we get $f(\breve{\boldsymbol{x}}) = \mathfrak{K}\mathbf{c}$. To simplify the optimization we assume that $\{t_j\}$ can only take values in the discrete space $\{\tilde{t}_1, \cdots, \tilde{t}_L\}$, that is, $t_j = \tilde{t}_i$, for some $i \in 1, 2, \cdots, L$ (e.g., a fixed finite fine grid), where we always choose $\tilde{t}_1 = 0$. Therefore, we can write $\left[f * \delta_{t_j}\right](\breve{\boldsymbol{x}}) = \widetilde{\mathcal{K}}_{t_j}^{\mathrm{T}}\mathbf{c}$, where $\widetilde{\mathcal{K}}_{t_j}$ is $\mathcal{K}_0(\breve{\boldsymbol{x}}, [(\breve{\boldsymbol{x}} - \tilde{t}_j) \mod T])$. Accordingly, Equation (28) is reduced to

$$\underset{\mathbf{c}\in\mathbb{R}^{\mathbb{N}}, t_1, \cdots, t_j \in \{\tilde{t}_i\}}{\operatorname{argmin}} \left\{ \sum_j \gamma_{js} \|\tilde{\boldsymbol{y}}^j - C^j \cdot \widetilde{\mathcal{K}}_{t_j}^{\mathrm{T}}\mathbf{c}\|^2 + \frac{1}{2}\mathbf{c}^{\mathrm{T}}\mathfrak{K}\mathbf{c} \right\}. \tag{30}$$

To solve this optimization, we follow a cyclic optimization approach where we alternate between steps of optimizing f and $\{t_j\}$ respectively,

- At step ℓ, optimize equation (30) with respect to $\{t_j\}$ given $\mathbf{c}^{(\ell)}$. Since $\mathbf{c}^{(\ell)}$ is known, it follows immediately that Equation (30) decomposes into M independent tasks, where for the jth task we need to find $t_j^{(\ell)}$ such that $C^j\widetilde{\mathcal{K}}_{t_j^{(\ell)}}^{\mathrm{T}}\mathbf{c}$ is closest to $\tilde{\boldsymbol{y}}^j$ under the Euclidean distance. A brute force search with time complexity $\boldsymbol{\Theta}(\mathbb{N}L)$ yields the optimal solution. If the time series are synchronously sampled (i.e. $C^j = \mathbb{I}, j = 1, \cdots, M$), this is equivalent to finding the shift τ corresponding the *cross-correlation*, defined as

  $$\mathcal{C}(\mathbf{u}, \mathbf{v}) = \max_{\tau} \langle \mathbf{u}, \mathbf{v}_{+\tau} \rangle \tag{31}$$

  where $\mathbf{u} = \mathfrak{K}\mathbf{c}$ and $\mathbf{v} = \tilde{\boldsymbol{y}}^j$ and $\mathbf{v}_{+\tau}$ refers to the vector $\mathbf{v}$ right shifted by τ positions, and where positions are shifted modulo $\mathbb{N}$. Furthermore, as shown by [21], if every $\boldsymbol{x}^j$ has regular time intervals, we can use the convolution theorem to find the same value in $\boldsymbol{\Theta}(\mathbb{N}\log\mathbb{N})$ time, that is

  $$t_j^{(\ell)} = \underset{\tau}{\operatorname{argmax}} \left(\mathscr{F}^{-1}\left[\mathscr{U} \cdot \widehat{\mathscr{V}}\right](\tau) \right) \tag{32}$$

  where $\mathscr{F}^{-1}[\cdot]$ denotes inverse Fourier transform, $\cdot$ indicates point-wise multiplication; $\mathscr{U}$ is the Fourier transform of $\mathbf{u}$ and $\widehat{\mathscr{V}}$ is the complex conjugate of the Fourier transform of $\mathbf{v}$.
- At step $\ell+1$, optimize equation (30) with respect to $\mathbf{c}^{(\ell+1)}$ given $t_1^{(\ell)}, \cdots, t_M^{(\ell)}$. For the jth task, since $t_j^{(\ell)}$ is known, denote $C^j\widetilde{\mathcal{K}}_{t_j^{(\ell)}}^{\mathrm{T}}$ as $\mathfrak{M}_j^{(\ell)}$. The regularized least square problem can be reformulated as

  $$\underset{\mathbf{c}\in\mathbb{R}^{\mathbb{N}}}{\operatorname{argmin}} \left\{ \sum_j \gamma_{js} \|\tilde{\boldsymbol{y}}^j - \mathfrak{M}_j^{(\ell)}\mathbf{c}\|^2 + \frac{1}{2}\mathbf{c}^T\mathfrak{K}\mathbf{c} \right\}. \tag{33}$$

Taking derivatives of Equation (33), we see that the new $\mathbf{c}^{(\ell+1)}$ value is obtained by solving the following linear system

$$-2\sum_j \gamma_{js} \cdot (\mathfrak{M}_j^{(\ell)})^{\mathrm{T}} \left(\tilde{\boldsymbol{y}}^j - \mathfrak{M}_j^{(\ell)} \cdot \mathbf{c}\right) + \mathfrak{K}\mathbf{c} = 0. \tag{34}$$

Obviously, each step decreases the value of the objective function and therefore the algorithm will converge.

Given the estimates of $\{\bar{f}_s\}, \{t_j\}$, the optimization for σ^2 is given by

$$\begin{aligned}\sigma^* = \underset{\sigma\in\mathbb{R}}{\operatorname{argmin}} \Bigg\{ & \sum_j n_j \log\sigma + \frac{1}{2\sigma^2}\sum_j \mathrm{Tr}(\mathbf{C}_j^g) \\ & \frac{1}{2\sigma^2}\sum_j\sum_s \gamma_{js}\left(\|\boldsymbol{y}^j - [\bar{f}_s^* * \delta_{t_j^*}](\boldsymbol{x}^j) - \boldsymbol{\mu}_{js}\|^2\right)\Bigg\}\end{aligned} \tag{35}$$

where $\{\bar{f}^*\}$ and $\{t_j^*\}$ are obtained from the previous optimization steps. Let $R = \sum_j \mathrm{Tr}(\mathbf{C}_j^g) + \sum_j\sum_s \gamma_{js}\left(\|\boldsymbol{y}^j - [\bar{f}_s^* * \delta_{t_j}](\boldsymbol{x}^j) - \boldsymbol{\mu}_{js}\|^2\right)$. Then it is easy to see that $(\sigma^*)^2 = R/\sum_j n_j$.

Learning the kernel for individual effect [16] have already shown how to optimize the kernel function in a similar context. Here we provide some of the details for completeness. If the kernel function $\mathcal{K}$ admits a parametric form with parameter θ, for example the RBF kernel

$$\mathcal{K}(x,y) = a\exp\left\{-\frac{\|x-y\|^2}{2s^2}\right\} \tag{36}$$

where $\theta = \{a, s\}$, then the optimization of the kernel $\mathcal{K}$ amounts to finding θ^* such that

$$\theta^* = \underset{\theta}{\operatorname{argmax}}\left\{ -\frac{1}{2}\sum_j \log|(\mathbf{K}_j;\theta)| - \frac{1}{2}\sum_j\sum_s \gamma_{js}\mathrm{Tr}\left((\mathbf{K}_j;\theta)^{-1}(\mathbf{C}_j^g + \boldsymbol{\mu}_{js}^g(\boldsymbol{\mu}_{js}^g)^{\mathrm{T}})\right)\right\}. \tag{37}$$

It is easy to see the gradient of the right hand side of Equation (37) is

$$-\frac{1}{2}\sum_j \mathrm{Tr}\left(\mathbf{K}_j\frac{\partial\mathbf{K}_j}{\partial\theta}\right) - \frac{1}{2}\sum_j\sum_s \gamma_{js}\mathrm{Tr}\left(\mathbf{K}_j^{-1}\frac{\partial\mathbf{K}_j}{\partial\theta}\mathbf{K}_j^{-1}(\mathbf{C}_j^g + \boldsymbol{\mu}_{js}^g(\boldsymbol{\mu}_{js}^g)^{\mathrm{T}})\right). \tag{38}$$

Therefore, any optimization method, e.g. conjugated gradients can be utilized to find the optimal parameters. Notice that given the inverse of kernel matrix $\{\mathbf{K}_j\}$, the computation of the derivative requires $\Theta(\sum n_j^2)$ steps. The parametric form of the kernel is a prerequisite to perform the regression task when examples are not sampled synchronously as in our development above.

If the data is synchronously sampled, for classification tasks we only need to find the kernel matrix $\mathbf{K}$ for the given sample points and the optimization problem can be rewritten as

$$\mathbf{K}^* = \underset{\mathbf{K}}{\operatorname{argmax}} \left\{ -\frac{1}{2}\sum_j \log|\mathbf{K}| - \frac{1}{2}\sum_j \sum_s \gamma_{js} \operatorname{Tr}\left(\mathbf{K}^{-1}(\mathbf{C}_j^g + \boldsymbol{\mu}_{js}^g(\boldsymbol{\mu}_{js}^g)^{\mathrm{T}})\right) \right\}. \tag{39}$$

Similar to maximum likelihood estimation for multivariate Gaussian distribution, the solution is

$$\mathbf{K}^* = \frac{1}{M}\sum_j \sum_s \gamma_{js}(\mathbf{C}_j^g + \boldsymbol{\mu}_{js}^g(\boldsymbol{\mu}_{js}^g)^{\mathrm{T}}). \tag{40}$$

In our experiments, we use both approaches where for the parametric form we use the RBF kernel as outlined above.

4.3 Algorithm Summary

The various steps in our algorithm and their time complexity are summarized in Algorithm 1.

Once the model parameters $\mathcal{M}$ are learned (or if they are given in advance), we can use the model to perform regression or classification tasks. The following summarizes the procedures used in our experiments.

- **Regression:** To predict a new sample point for an existing task (task j) we calculate its most likely cluster assignment z_j and then predict the y value based on this cluster. Concretely, z_j is determined by

$$z_j = \underset{s=\{1,\cdots,k\}}{\operatorname{argmax}} \left[\Pr(z_j = s|\boldsymbol{x}^j, \boldsymbol{y}^j; \mathcal{M})\right] \tag{41}$$

  and given a new data point x, the prediction y is given by

$$y = \left[\bar{f}_{z_j} * \delta_{t_j}\right](x) + \tilde{f}^j(x).$$

- **Classification:** For classification, we get a new time series and want to predict its label. Recall from Section 3, Equation (11) that we learn a separate model for each class and predict using

$$o = \underset{\ell=\{1,\cdots,L\}}{\operatorname{argmax}} \Pr(\boldsymbol{y}|\boldsymbol{x}; \mathrm{M}_\ell) \Pr(\ell).$$

  In this context, $\Pr(\ell)$ is estimated by the frequencies of each class and the likelihood portion is given by first finding the best time shift t for the new time series and then calculating the likelihood according to

$$\Pr(\boldsymbol{y}|\boldsymbol{x}; M_\ell) = \sum_z \Pr(z|\mathcal{M}_\ell) \Pr(\boldsymbol{y}|z, \boldsymbol{x}; \mathcal{M}_\ell) \tag{42}$$

  where $\mathcal{M}_\ell$ is the learned parameter set and the second term is calculated via Equation (16).

Algorithm 1 EM algorithm for Shift-invariant GMT

Initialize $\{f_s^{(0)}\}, \{t_j^{(0)}\}, \alpha^{(0)}$ and $\mathcal{K}^{(0)}$.
repeat
  Calculate $\mathbf{K}_j^{(t)}$ according to $\boldsymbol{x}^j, \mathcal{K}^{(t-1)}$. The time complexity for constructing kernel are $\Theta(\sum n_j^2)$ and $\Theta(1)$ in parametric and nonparametric case respectively.
  Calculate γ_{js} according to Equation (22). For each task, we need to invert the covariance matrix in the marginal distribution and then calculate the likelihood, thus the time complexity is $\Theta(\sum n_j^3)$.
  for all s such that $0 \leq s \leq k$ **do**
    Update $\alpha^{(t)}$ such that $\alpha_s^{(t)} = \sum_j \gamma_{js}/M$.
    repeat
      Update $\{t_j\}$ w.r.t. cluster s such that $t_j \in \{\tilde{t}_1, \cdots, \tilde{t}_L\}$ and minimize $\|\tilde{\boldsymbol{y}}^j - C^j \cdot \widetilde{\mathcal{K}}_{t_j}^{\mathrm{T}} \mathbf{c}_s^{(0)}\|^2$. The time complexity is $\Theta(L\mathbb{IN})$ as discussed above.
      Update $\mathbf{c}_s^{(t+1)}$ by solving linear system Equation (34), which requires $\Theta(\mathbb{I}^3)$.
    until converges **or** reach the iteration limit
  end for
  Update $\sigma^{(t+1)}$ according to Equation (35).
  Update the parameters of the kernel or the kernel matrix directly via optimizing Equation (37) or using the closed-form solution Equation (40) for $\mathbf{K}$. In the former case, a gradient based optimizer can be used with time complexity $\Theta(\sum n_j^2)$ for each iteration; while in the later case, the estimation only requires $\Theta(kM\mathbb{IN})$.
until converges or reach the iteration limit

5 Infinite mixture of Gaussian processes

In this section we develop an extension of the model removing the assumption that the number of centers k is known in advance.

5.1 Dirichlet process basics

We start by reviewing basic concepts for Dirichlet processes. Suppose we have i.i.d. data such that

$$x_1, x_2, \cdots, x_n \sim \mathcal{F}$$

where $\mathcal{F}$ is an unknown distribution that needs to be inferred from $\{x_i\}$. A Parametric Bayesian approach assumes $\mathcal{F}$ is given by a parametric family $\mathcal{F}_\theta$ and the parameters θ follow a certain distribution that comes from our prior belief. However, this assumption has limitations both in the scope and the type of inferences that can be performed. Instead, nonparametric Bayesian approach places a prior distribution on the distribution $\mathcal{F}$ directly. The Dirichlet process (DP) is used for such purpose. The DP is parameterized by a base distribution G_0 and a positive scaling parameter (or concentration parameter) α. A random

measure G is distributed according to a DP with base measure G_0 and scaling parameter α if for all finite measurable partitions $\{B_i\}, i = 1, \ldots, k$,

$$(G(B_1), G(B_2), \cdots, G(B_k)) \sim \text{Dir}(\alpha G_0(B_1), \alpha G_0(B_2), \cdots, \alpha G_0(B_k))$$

where $\text{Dir}(\cdot)$ is the Dirichlet distribution. It is known that G is almost surely a discrete measure.

The Dirichlet process mixture model extends this setting, where the DP is used as a nonparametric prior in a hierarchical Bayesian specification. More precisely,

$$\begin{aligned} G|\{\alpha, G_0\} &\sim \mathcal{DP}(\alpha, G_0) \\ \eta_n|G &\sim G \qquad n = 1, 2, \ldots \\ x_n|\eta_n &\sim f(x_n|\eta_n) \end{aligned}$$

where f is some probability density function that is parameterized by η. Data generated from this model can be naturally partitioned according to the distinct values of the parameter η_n. Hence, the DP mixture can be interpreted as a mixture model where the number of mixtures is flexible and grows as the new data is observed. Alternatively, we can view the infinite mixture model as the limit of the finite mixture model. Consider the Bayesian finite mixture model with a symmetric Dirichlet distribution as the prior of the mixture proportions. When the number of mixtures $k \to \infty$, the Dirichlet distribution becomes a Dirichlet process [**?**, see]]neal2000markov.

[22] provides a more explicit construction of the DP which is called the *stick-breaking construction* (SBC). Given $\{\alpha, G_0\}$, we have two collections of random variables $V_i \sim \text{Beta}(1, \alpha)$ and $\eta_i^* \sim G_0, \quad i = \{1, 2, \cdots, \}$. The SBC of G is

$$\pi_i(\mathbf{v}) = v_i \prod_{j=1}^{i-1}(1 - v_j)$$

$$G = \sum_{i=1}^{\infty} \pi_i(\mathbf{v})\delta_{\eta_i^*}.$$

If we set $v_K = 1$ for some K, then we get a truncated approximation to the DP

$$G = \sum_{i=1}^{K} \pi_i(\mathbf{v})\delta_{\eta_i^*}.$$

[23] shows that when selecting the truncation level K appropriately, the truncated DP behaves very similarly to the original DP.

5.2 The DP-GMT Model and Inference Algorithm

In this section, we extend our model by modeling the mixture proportions using a DP prior. The plate graph is shown in Figure 2. Under the SBC, the generative process is as follows

1. Draw $v_s|\alpha \sim \text{Beta}(1,\alpha)$, $s = \{1,2,\ldots\}$
2. Draw $\bar{f}_s|\mathcal{K}_0 \sim \exp\left\{-\frac{1}{2}\|\bar{f}_s\|^2_{\mathcal{H}_0}\right\}$, $s = \{1,2,\ldots\}$
3. For the jth time series
   (a) Draw $z_j|\{v_1, v_2, \ldots\} \sim \text{Discrete}(\pi(\mathbf{v}))$, where $\pi_s(\mathbf{v}) = v_s \prod_{i=1}^{s-1}(1-v_i)$;
   (b) Draw $\tilde{f}^j|\mathcal{K} \sim \exp\left\{-\frac{1}{2}\|\tilde{f}^j\|^2_{\mathcal{H}}\right\}$;
   (c) Draw $\boldsymbol{y}^j|z_j, f^j, \boldsymbol{x}^j, t_j, \sigma^2 \sim \mathcal{N}\left(f^j(\boldsymbol{x}^j), \sigma^2\mathbb{I}\right)$, where $f^j = \bar{f}_{z_j} * \delta_{t_j} + \tilde{f}^j$.

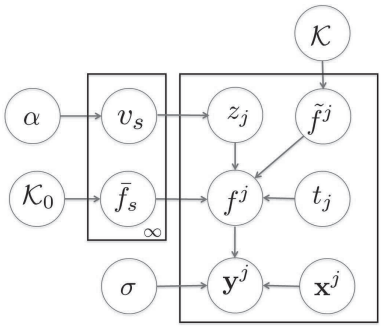

Fig. 2. DPGMT: Plate graph

In this model, the concentration parameter α is assumed to be known. As in Equation (12), the inference task is to find the MAP estimates of the parameter set $\mathcal{M} = \{\{\bar{f}_s\}, \{t_j\}, \sigma^2, \mathcal{K}\}$. Notice that in contrast with the previous model, the mixture proportion are not estimated here. To perform the inference, we must consider another set of hidden variables $\mathbf{v} = \{v_i\}$ in addition to $\mathbf{f}$ and $\mathbf{z}$. However, calculating the posterior of the hidden variables is intractable, thus the variational EM algorithm [**?**, e.g.,]]bishop2006pattern is used to perform the approximate inference. The algorithm can be summarized as follows:

– **Variational E-Step** Choose a family $\mathcal{G}$ of variational distributions $q(\mathbf{f}, \mathbf{v}, \mathbf{z})$ and find the distribution q^* that minimizes the Kullback-Leibler (**KL**) divergence between the posterior distribution and the proposed distribution given the current estimate of parameters, i.e.

$$q^*(\mathbf{f}, \mathbf{v}, \mathbf{z}; \mathcal{M}^g) = \underset{q \in \mathcal{G}}{\text{argmin}}\ \mathbf{KL}\left(\Pr(\mathbf{f}, \mathbf{v}, \mathbf{z}|\mathcal{X}, \mathcal{Y}; \mathcal{M}^g)||q(\mathbf{f}, \mathbf{v}, \mathbf{z})\right) \tag{43}$$

where

$$\mathbf{KL}\left(\Pr(\mathbf{f}, \mathbf{v}, \mathbf{z}|\mathcal{X}, \mathcal{Y}; \mathcal{M}^g)||q(\mathbf{f}, \mathbf{v}, \mathbf{z})\right) = \int \log\left[\frac{\Pr(\mathbf{f}, \mathbf{v}, \mathbf{z}|\mathcal{X}, \mathcal{Y}; \mathcal{M}^g)}{q(\mathbf{f}, \mathbf{v}, \mathbf{z})}\right] \mathrm{d}q(\mathbf{f}, \mathbf{v}, \mathbf{z}).$$

– **Variational M-Step** Optimize the parameter set $\mathcal{M}$ such that

$$\mathcal{M}^* = \underset{\mathcal{M}}{\text{argmax}}\ Q(\mathcal{M}, \mathcal{M}^g)$$

where

$$Q(\mathcal{M}, \mathcal{M}^g) = \mathbb{E}_{q^*(\mathbf{z},\mathbf{f},\mathbf{v};\mathcal{M}^g)} \left[\log \left\{ \Pr(\mathcal{Y}, \mathbf{f}, \mathbf{z}, \mathbf{v} | \mathcal{X}; \mathcal{M}) \times \Pr[\{\bar{f}_s\}; \mathcal{K}_0] \right\} \right]. \tag{44}$$

Variational E-Step. For the variational distribution $q()$ we use the *mean field approximation* [24]. That is we assume a factorized distribution for disjoint group of random variables. This results in an analytic tractable optimization problem. In addition, following [25], we approximate the distribution over $\mathbf{v}$ using a truncated stick-breaking representations, where for a fix T, $q(v_T = 1) = 1$ and therefore $\pi_s(\mathbf{v}) = 0,\ s > T$. In this paper, we fix the truncation level T while in general it can also be treated as a variational parameter. Concretely, we propose the following factorized family of variational distributions over the hidden variables $\{\mathbf{f}, \mathbf{v}, \mathbf{z}\}$:

$$q(\mathbf{f}, \mathbf{v}, \mathbf{z}) = \prod_{s=1}^{T-1} q_s(v_s) \prod_{j=1}^{M} q_j(f_j, z_j). \tag{45}$$

Note that we do not assume any parametric form for $\{q_s, q_j\}$ and our only assumption is that the distribution factorizes into independent components. To optimize Equation (43), recall the following result from [**?**,]Chapter 8]bishop2006pattern:

Lemma 1. *Suppose we are given a probabilistic model with a joint distribution* $\Pr(\mathbf{X}, \mathbf{Z})$ *over* $\mathbf{X}, \mathbf{Z}$ *where* $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$ *denote the observed variables and all the parameters and hidden variables are* $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_M\}$. *Assume the distribution of* $\mathbf{Z}$ *has the following form:*

$$q(\mathbf{Z}) = \prod_{i}^{M} q_i(z_i).$$

Then, the ***KL*** *divergence between the posterior distribution* $\Pr(\mathbf{Z}|\mathbf{X})$ *and* $q(\mathbf{Z})$ *is minimized and the optimal solution* $q_j^*(\mathbf{z}_j)$ *is given by*

$$q_j^*(\mathbf{z}_j) \propto exp\left(\mathbb{E}_{i \neq j}\left[\log \Pr(\mathbf{X}, \mathbf{Z})\right]\right)$$

where $\mathbb{E}_{i \neq j}[\cdots]$ *denotes the expectation w.r.t.* $q()$ *over all* $Z_j,\ j \neq i$.

From the graphical model in Figure 2, the joint distribution of $\Pr(\mathcal{Y}, \mathbf{f}, \mathbf{v}, \mathbf{z}|\mathcal{X}; \mathcal{M}^g)$ can be written as:

$$\begin{aligned}
\Pr(\mathcal{Y}, \mathbf{f}, \mathbf{v}, \mathbf{z}|\mathcal{X}) &= \Pr(\mathcal{Y}|\mathcal{X}, \mathbf{f}, \mathbf{z}) \Pr(\mathbf{z}|\mathbf{v}) \Pr(\mathbf{f}|\mathcal{X}) \Pr(\mathbf{v}|\alpha) \\
&= \prod_j \Pr(\boldsymbol{y}^j|\boldsymbol{x}^j, \mathbf{f}_j, z_j) \prod_j \Pr(z_j|\mathbf{v}) \prod_j \Pr(\mathbf{f}_j|\boldsymbol{x}^j) \prod_s \Pr(v_s|\alpha).
\end{aligned}$$

Equivalently,

$$\begin{aligned}
\log \Pr(\mathcal{Y}, \mathbf{f}, \mathbf{v}, \mathbf{z}|\mathcal{X}) = &\sum_j \log \Pr(\boldsymbol{y}^j|\boldsymbol{x}^j, \mathbf{f}_j, z_j) \\
&+ \sum_j \log \Pr(z_j|\mathbf{v}) + \sum_j \log \Pr(\mathbf{f}_j|\boldsymbol{x}^j) + \sum_s \log \Pr(v_s|\alpha).
\end{aligned}$$

First we consider the distribution of $q_s(\mathbf{v})$. Following [25], the second term can be expanded as

$$\log \Pr(z_j|\mathbf{v}) = \sum_{t=1}^{T} \mathbb{1}_{\{z_j>t\}} \log(1-v_t) + \mathbb{1}_{\{z_j=t\}} \log v_t \tag{46}$$

where $\mathbb{1}$ is the indicator function. Therefore, using the lemma above and denoting $\mathbf{v}\backslash v_s$ by $\mathbf{v}_{-s}$, we have

$$\begin{aligned}
\log q_s(v_s) &\propto \mathbb{E}_{\mathbf{z},\mathbf{f},\mathbf{v}_{-s}} [\log \Pr(\mathcal{Y}, \mathbf{f}, \mathbf{v}, \mathbf{z}|\mathcal{X})] \\
&= \sum_j \left(\mathbb{E}_{\mathbf{z},\mathbf{f},\mathbf{v}_{-s}} [\mathbb{1}_{\{z_j>s\}}] \log(1-v_s) + \mathbb{E}_{\mathbf{z},\mathbf{f},\mathbf{v}_{-s}} [\mathbb{1}_{\{z_j=s\}}] \log v_s \right) + \log \Pr(v_s|\alpha) + \text{constant} \\
&= \sum_j \left(q(z_j > s) \log(1-v_s) + q(z_j = s) \log v_s \right) + \log \Pr(v_s|\alpha) + \text{constant}
\end{aligned}$$

Recalling that the prior is given by $\text{Beta}(1, \alpha)$ we see that the distribution of $q_s(v_s)$ is

$$q_s(v_s) \propto v_s^{\sum_j q(z_j=s)} (1-v_s)^{\alpha + \sum_j \sum_{l=s+1}^{T} q(z_j=l) - 1}.$$

Observing the form of $q_s(v_s)$, we can see that it is a Beta distribution and $q_t(v_t) \sim \text{Beta}(\gamma_{t,1}, \gamma_{t,2})$ where

$$\gamma_{t,1} = 1 + \sum_j q(z_j = t)$$

$$\gamma_{t,2} = \alpha + \sum_j \sum_{l=s+1}^{T} q(z_j = l).$$

We next consider $q_j(\mathbf{f}_j, z_j)$. Notice that we can always write $q_j(\mathbf{f}_j, z_j) = q_j(\mathbf{f}_j|z_j) q_j(z_j)$. Denote $h(z_j) = \mathbb{E}_{\mathbf{v}} [\log \Pr(z_j|\mathbf{v})]$, then again using the lemma above we have

$$\begin{aligned}
q_j(\mathbf{f}_j|z_j) q_j(z_j) &\propto e^{h(z_j)} \Pr(\boldsymbol{y}^j|\mathbf{x}^j, \mathbf{f}_j, z_j) \Pr(\mathbf{f}_j|\boldsymbol{x}^j) \\
&= e^{h(z_j)} \Pr(\boldsymbol{y}^j, \mathbf{f}_j|\boldsymbol{x}^j, z_j) \\
&\propto e^{h(z_j)} \Pr(\boldsymbol{y}^j|\boldsymbol{x}^j, z_j) \Pr(\mathbf{f}_j|\boldsymbol{x}^j, \boldsymbol{y}^j, z_j) \\
&\propto \left[\underbrace{e^{h(z_j)} \Pr(\boldsymbol{y}^j|\boldsymbol{x}^j, z_j)}_{q_j(z_j)} \right] \left[\underbrace{\Pr(\mathbf{f}_j|\boldsymbol{x}^j, \boldsymbol{y}^j, z_j)}_{q_j(\mathbf{f}_j|z_j)} \right].
\end{aligned}$$

The equality in the second line holds because $\Pr(\mathbf{f}_j|\boldsymbol{x}^j) = \Pr(\mathbf{f}_j|\boldsymbol{x}^j, z_j)$; their distributions become coupled when conditioned on the observations $\boldsymbol{y}^j$, but without such observations they are independent. Therefore the left term yields

$$q_j(z_j) \propto e^{h(z_j)} \Pr(\boldsymbol{y}^j|\boldsymbol{x}^j, z_j)$$

where $\Pr(\boldsymbol{y}^j|\boldsymbol{x}^j, z_j)$ is given by Equation (16). The value of $h(z_j)$ can be calculated using Equation (46):

$$\log \Pr(z_j = s|\mathbf{v}) = \sum_{t=1}^{s-1} \log(1 - v_t) + \log v_s$$
$$h(z_j = s) = \mathbb{E}_{v_s}[\log v_s] + \sum_{i=1}^{s-1} \mathbb{E}_{v_i}\left[\log(1 - v_i)\right]$$

where

$$\begin{aligned} \mathbb{E}_{v_t}[\log v_t] &= \Psi(\gamma_{i,1}) - \Psi(\gamma_{i,1} + \gamma_{i,2}) \\ \mathbb{E}_{v_i}\left[\log(1 - v_i)\right] &= \Psi(\gamma_{i,2}) - \Psi(\gamma_{i,1} + \gamma_{i,2}). \end{aligned} \tag{47}$$

Consequently, $q_j(z_j)$ has the following form

$$q_j(z_j = t) \propto \exp\left\{\mathbb{E}_{v_t}[\log v_t] + \sum_{i=1}^{t-1} \mathbb{E}_{v_i}\left[(1 - v_i)\right]\right\} \times \mathcal{N}\left(\overline{\mathbf{f}}^j, \mathbf{K}_j^g + \sigma^2\mathbb{I}\right). \tag{48}$$

Note that this is the same form as in Equation (15) of the previous model where $\Pr(z_j; \mathcal{M}^g)$ is replaced by $e^{h(z_j=t)}$.

Given z_j, $q_j(\mathbf{f}_j|z_j)$ is identical to Equation (17) and leads to the conditional distribution such that

$$q_j(\mathbf{f}_j|z_j) \propto \Pr(\boldsymbol{y}^j|\boldsymbol{x}^j, \mathbf{f}_j, z_j) \Pr(\mathbf{f}_j; \boldsymbol{x}^j)$$

which is the posterior distribution under GP regression and thus is exactly the same form as in the previous model.

Variational M-Step. Denote $\widetilde{Q}$ as the expectation of the complete data log likelihood w.r.t. the hidden variables. Then as in Equation (20), we have

$$\begin{aligned} \widetilde{Q} &= \mathbb{E}_{q(\mathbf{v})}\mathbb{E}_{q(\mathbf{z})}\mathbb{E}_{q(\mathbf{f}|\mathbf{z})} \log\left(\prod_j \prod_s \left[\pi_s(\mathbf{v}) \Pr(\boldsymbol{y}^j|\mathbf{f}_j, z_j = s; \mathcal{M}) \Pr(\mathbf{f}_j; \mathcal{M})\right]^{z_{js}}\right) \\ &= \mathbb{E}_{\mathbf{v}}\left[\sum_{\mathbf{z}} q(\mathbf{z})\left\{\sum_j \sum_s z_{js} \cdot \int dq(\mathbf{f}_j|z_j = s)\right.\right. \\ &\qquad \left.\left. \times \log\left[\pi_s(\mathbf{v}) \Pr(\boldsymbol{y}^j|\mathbf{f}_j, z_j = s; \mathcal{M}) \Pr(\mathbf{f}_j; \mathcal{M})\right]\right\}\right] \\ &= \sum_{\mathbf{z}} q(\mathbf{z})\left\{\sum_j \sum_s z_{js} \cdot \int dq(\mathbf{f}_j|z_j = s)\right. \\ &\qquad \left. \times \log\left[\Pr(\boldsymbol{y}^j|\mathbf{f}_j, z_j = s; \mathcal{M}) \Pr(\mathbf{f}_j; \mathcal{M})\right]\right\} + \mathbb{E}_{\mathbf{v}}\left[\sum_j \sum_s \log \pi_s(\mathbf{v})\right]. \end{aligned} \tag{49}$$

Notice that $\mathbb{E}_{\mathbf{v}}\left[\sum_j \sum_s \log \pi_s(\mathbf{v})\right]$ is a constant w.r.t. the parameters of $\mathcal{M}$ and can be dropped in the optimization. Thus, following the same derivation as in the GMT model, we have the form of the Q function as

$$\begin{aligned} Q(\mathcal{M}, \mathcal{M}^g) = &-\frac{1}{2}\sum_s \|\bar{f}_s\|^2_{\mathcal{H}_0} - \sum_j n_j \log \sigma \\ &- \frac{1}{2\sigma^2}\sum_j \sum_s \gamma_{js} \mathbb{E}_{\{q(\mathbf{f}_j|z_j=s)\}}\left[\|\boldsymbol{y}^j - [\bar{f}_s * \delta_{t_j}](\boldsymbol{x}^j) - \mathbf{f}_j\|^2\right] \\ &+ \sum_j \sum_s \gamma_{js} \mathbb{E}_{\{q(\mathbf{f}_j)\}}\left[\log \Pr(\mathbf{f}_j; \mathcal{M})\right]. \end{aligned} \tag{50}$$

where γ_{js} is given by Equation (22). Now because the $q_j(z_j)$ and $q_j(f_j|z_j)$ have exactly the same form as before (except $\Pr(z_j; \mathcal{M}^g)$ is replaced by Equation (48)), the previous derivation of the **M-Step** w.r.t. the parameter set $\mathcal{M}$ still holds.

To summarize, the algorithm is the same as Algorithm 1 except that

- we drop step 6,
- we add a step between steps 3 and 4 calculating $\gamma_{i,1}$ and $\gamma_{i,2}$ using Equation (47),
- step 4 calculating Equation (22) uses Equation (48) instead of Equation (15).

6 Experiments

Our implementation of the algorithm makes use of the gpml package [26] and extends it to implement the required functions. The EM algorithm is restarted 5 times and the function that best fits the data is chosen. The EM algorithm stops when difference of the log-likelihood is less than 10e-5 or at a maximum of 200 iterations.

6.1 Regression on Synthetic data

In the first experiment, we demonstrate the performance of our algorithm on a regression task with artificial data. We generated the data following Assumption 1 under a mixture of three Gaussian processes. More precisely, each $\bar{f}_s(x), s = 1, 2, 3$ is generated on the interval $[-50, 50]$ from a Gaussian process with covariance function

$$\operatorname{cov}[\bar{f}_s(t_1), \bar{f}_s(t_2)] = e^{-\frac{(t_1-t_2)^2}{25}}, \quad s = 1, 2, 3.$$

The individual effect $\tilde{f}_j$ is sampled via a Gaussian process with the covariance function

$$\operatorname{cov}[\tilde{f}_j(t_1), \tilde{f}_j(t_2)] = 0.2e^{-\frac{(t_1-t_2)^2}{16}}.$$

Then the hidden label z_j is sampled from a discrete distribution with the parameter $\alpha = [0.5, 0.5]$. The vector $\breve{\boldsymbol{x}}$ consists of 100 samples on $[-50, 50]$[5]. We fix a sample size N, each $\boldsymbol{x}^j$ includes N randomly chosen points from $\{\breve{x}_1, \cdots, \breve{x}_{100}\}$ and the observation $f^j(\boldsymbol{x}^j)$ is obtained as $(f_{z_j} + \tilde{f}_j)(\boldsymbol{x}^j)$. In the experiment, we vary the individual sample length N from 5 to 50. Finally, we generated 50 random tasks with the observation $\boldsymbol{y}^j$ for task j given by

$$\boldsymbol{y}^j \sim \mathcal{N}(f^j(\boldsymbol{x}^j), 0.01 \times \mathbb{I}), \quad j = 1, \cdots, 50.$$

The methods compared here include

1. **Single-task learning procedure (ST)**, where each f^j is estimated only using $\{\boldsymbol{x}_i^j, \boldsymbol{y}_i^j\}, i = 1, 2, \cdots, N$.
2. **Single center mixed-effect multi-task learning (SCMT)**, amounts to the mixed-effect model [9] where one average function $\bar{f}$ is learned from $\{\boldsymbol{x}^j, \boldsymbol{y}^j\}, j = 1, \cdots, 50$ and $f^j = \bar{f} + \tilde{f}^j, j = 1, \cdots, 50$.
3. **Grouped mixed-effect model (GMT)**, the proposed method with number of clusters fixed to be the true model order.
4. **Dirichlet process Grouped mixed-effect model (DP-GMT)**, the infinite mixture extension of the proposed model.
5. **"Cheating" grouped fixed-effect model (CGMT)**, which follows the same algorithm as the grouped mixed-effect model but uses the true label z_j instead of their expectation for each task j. This serves as an upper bound for the performance of the proposed algorithm.

All algorithms (except for **ST** which does not estimate the kernel of the individual variations) use the same method to learn the kernel of the individual effects, which is assumed to have the form

$$\mathrm{cov}[\tilde{f}_j(t_1), \tilde{f}_j(t_2)] = ae^{-\frac{(t_1 - t_2)^2}{s^2}}.$$

The Root Mean Square Error (RMSE) for the four approaches is reported. For task j, the RMSE is defined as

$$\mathrm{RMSE}_j = \sqrt{\frac{1}{100}\|f(\breve{\boldsymbol{x}}) - f^j(\breve{\boldsymbol{x}})\|^2}$$

where f is the learned function and RMSE for the data set is the mean of $\{\mathrm{RMSE}_j\}, j = 1, \cdots, 50$. To illustrate the results qualitatively, we first plot in Figure 3 the true and learned functions in one trial. The left/center/right column illustrates some task that is sampled from group effect $\bar{f}_1$, $\bar{f}_2$ and $\bar{f}_3$. It it easy to see that, as expected, the tasks are poorly estimated under ST due the sparse sampling. The SCMT performs better than ST but its estimate is poor in areas where the three centers disagree. The estimates of GMT are much closer to the true function.

Figure 4 shows a comparison of the algorithms for 50 random data sets under the above setting when N equals 5. We see that GMT with the correct model

[5] The samples are generated via Matlab command: linspace(-50,50,100).

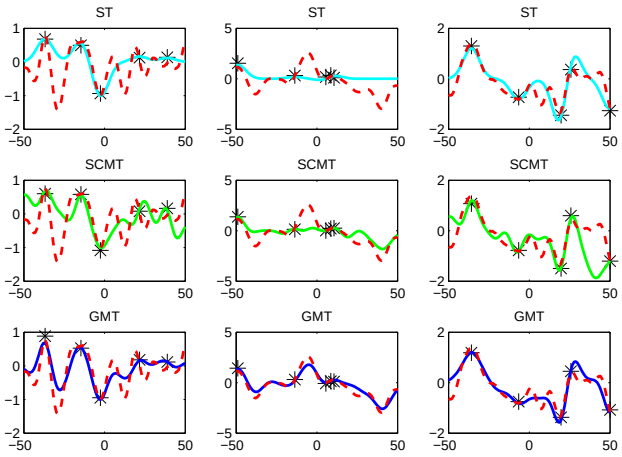

Fig. 3. Simulated data: Comparison of the estimated function between single, multi-task and grouped multi-task. The red dotted line is the reference true function.

order $k = 3$ almost always performs as well as its upper bound, illustrating that it recovers the correct membership of each task. On only three data sets, our algorithm is trapped in a local maximum yielding performance similar to SCMT and ST. Figure 5 shows the RMSE for increasing values of N for the same experimental setup. From the plot we can draw the conclusion that the proposed method works much better than SCMT and ST when the number of samples is less than 30. As the number of samples for each task increases, all methods are improving, but the proposed method always outperforms SCMT and ST in our experiments. Finally, all algorithms converge to almost the same performance level where observations in each task are sufficient to recover the underlying function. Finally, Figure 5 also includes the performance of the DP-GMT on the same data. The truncation level of the Dirichlet process is 10 and the concentration parameter α is set to be 1. As we can see the DP-GMT is not distinguishable from the GMT (which has the correct k), indicating that the model selection is successful in this example.

6.2 Classification on Astrophysics data

The concrete application motivating this research is the classification of stars into several meaningful categories from the astronomy literature. Classification is an important step within astrophysics research, as evidenced by published catalogs such as OGLE [27] and MACHO [28, 29]. However, the number of stars in such surveys is increasing dramatically. For example Pan-STARRS [30] and LSST [31] collect data on the order of hundreds of billions of stars. Therefore, it is

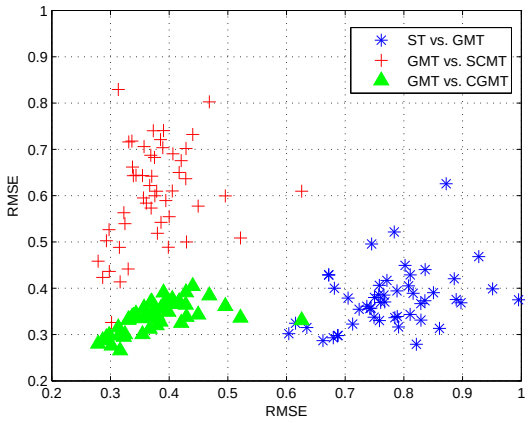

Fig. 4. Simulated data: Comparison between single, multi-task, and grouped multi-task when sample size is 5. The figure gives 3 pairwise comparison. The Blue stars denote ST vs. GMT: we can see the GMT is better than ST since the stars are concentrated on the lower right. Similarly, the plot of red pluses demonstrates the advantage of GMT over SCMT and the plot of green triangles shows that the algorithm behaves almost as well as its upper bound.

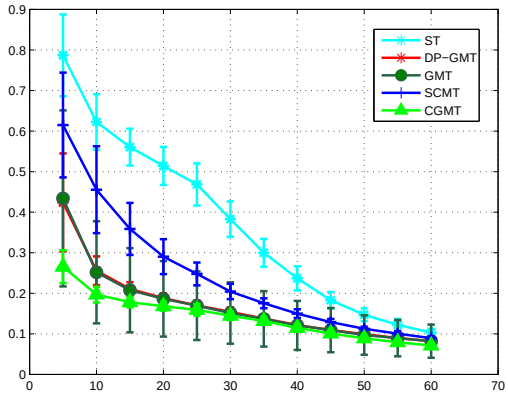

Fig. 5. Simulated data: Performance comparison of single, multi-task, and grouped multi-task, and DP grouped multi-task as a function of the number of samples per task.

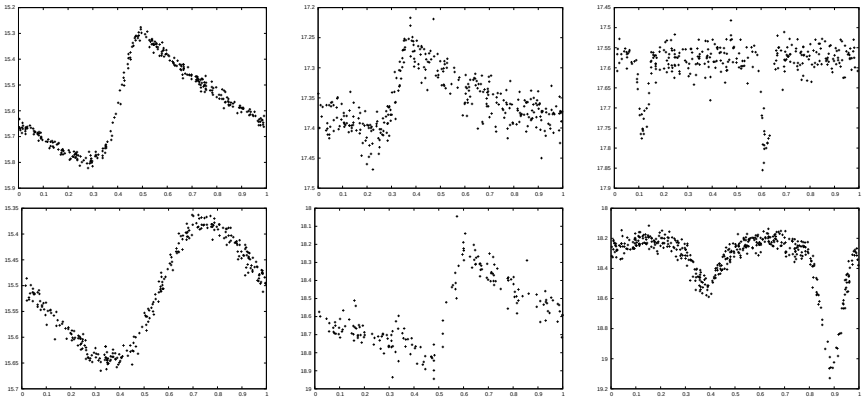

Fig. 6. Examples of light curves of periodic variable stars. Each column shows two stars of the same type. Left: Cepheid, middle: RR Lyrae, right: eclipsing binary.

desirable to apply state-of-art machine learning techniques to enable automatic processing for astrophysics data classification.

The data from star surveys is normally represented by time series of brightness measurements, based on which they are classified into categories. Stars whose behavior is periodic are especially of interest in such studies. Figure 6 shows several examples of such time series generated from the three major types of periodic variable stars: Cepheid, RR Lyrae, and Eclipsing Binary. In our experiments only stars of these classes are present in the data, and the period of each star is given.

From Figure 6, it can be noticed that there are two main characteristics of this data set:

- The time series are not phase aligned, meaning that the light curves in the same category share a similar shape but with some unknown shift.
- The time series are non-synchronously sampled and each light curve has different number of samples and sampling times.

We run our experiment on the OGLEII data set [32]. This data set consists of 14087 time series from periodic variable stars with 3425 Cepheids, 3390 EBs and 7272 RRLs. We use the time series measurements in the I band [32]. We perform several experiments with this data set to explore the potential of the proposed method. In previous work with this dataset [33] developed a kernel for periodic time series and used it with SVM to obtain good classification performance. We use the results of [33] as our baseline.[6]

[6] [33] used additional features, in addition to time series itself, to improve the classification performance. Here we focus on results using the time series only. Extensions to add such features to our model are orthogonal to the theme of the paper and we therefore leave them to future work in the context of the application.

	UP + GMM	GMT	UP + 1-NN	K + SVM
RESULTS	0.956 ± 0.006	0.952 ± 0.005	0.865 ± 0.006	0.947 ± 0.005

Table 1. Accuracies with standard deviations reported on OGLEII dataset.

Classification using dense-sampled time series In the first experiment, the time series are smoothed using a simple average filter, re-sampled to 50 points via linear-interpolation and normalized to have mean 0 and standard deviation of 1. Therefore, the time series are synchronously sampled in the pre-processing. We compare our method to Gaussian mixture model (GMM) and 1-Nearest Neighbor (1-NN). These two approaches are performed on the time series processed by Universal phasing (UP), which uses the method from [21] to phase each time series according to the sliding window on the time series with the maximum mean. We use a sliding window size of 5% of the number of original points; the phasing takes place after the pre-processing explained above. We learn a separate model for each class and for each class the model order for GMM and GMT is set to be 15.

We run 10-fold cross-validation (CV) over the entire data set and the results are shown in Table 1. We see that when the data is densely and synchronously sampled, the proposed method performs similar to the GMM, and they both outperform the kernel based results of [33]. The similarity of the GMM and the proposed method under these experimental conditions is not surprising. The reason is that when the time series are synchronously sampled, aside from the difference of phasing, finding the group effect functions is reduced to estimating the mean vectors of the GMM. In addition, learning the kernel in the non-parametric approach is equivalent to estimating the covariance matrix of the GMM. More precisely, assuming all time series are phased (that is, $t_j = 0$ for all j), the following results hold:

1. By placing a flat prior on the group effect function $\bar{f}_s, s = 1, \cdots, k$, or equivalently setting $\|\bar{f}_s\|^2_{\mathcal{H}_0} = 0$, Equation (28) is reduced to finding a vector $\mu_s \in \mathbb{N}$ that minimizes $\sum_j \gamma_{js} \|\tilde{\boldsymbol{y}}^j - \mu_s\|^2$. Therefore, we obtain $\bar{f}_s = \mu_s = \sum_j \gamma_{js} \tilde{\boldsymbol{y}}_j / \sum_j \gamma_{js}$, which is exactly the mean of the sth cluster during the iteration of EM algorithm under the GMM setting.

2. The kernel $\mathbf{K}$ is learned in a non-parametric way. For the GP regression model, we see that considering noisy observations is essentially equivalent to considering non-noisy observations, but slightly modifying the model by adding a diagonal term on the covariance function for $\mathbf{f}_j$. Therefore, instead of estimating $\mathbf{K}$ and σ^2, it is convenient to put these two terms together, forming $\widehat{\mathbf{K}} = \mathbf{K} + \sigma^2 \mathbb{I}$. In other words, we add a σ^2 term to the variance of $\mathbf{f}_j$ and remove it from $\boldsymbol{y}^j$ which becomes deterministic. In this case, comparing to the derivation in Equation (16)—(19) we have $\mathbf{f}_j = \boldsymbol{y}^j - \bar{\mathbf{f}}^j$ and $\mathbf{f}_j$ is determined given z_j. Comparing to Equation (17) we have the posterior mean $\boldsymbol{\mu}^g_{js} = \widehat{\mathbf{K}}\widehat{\mathbf{K}}^{-1}(\boldsymbol{y}^j - \mu_s) = \boldsymbol{y}^j - \mu_s$ and the posterior covariance matrix $\mathbf{C}^g_j$ vanishes. Applying these values in Equation (40) we get $\widehat{\mathbf{K}} = \frac{1}{M} \sum_j \sum_s \gamma_{js} (\boldsymbol{y}^j - \mu_s)(\boldsymbol{y}^j - \mu_s)^{\mathrm{T}}$. In the standard

EM algorithm for the GMM, this is equal to the estimated covariance matrix when all k clusters are assumed to have the same variance.

Accordingly, when time series are synchronously sampled, the proposed model can be viewed as an extension of the Phased K-means [10]. The Phased K-means (PKmeans) re-phases the time series before the similarity calculation and updates the centroids using the phased time series. Therefore, with shared covariance matrix, our model is a shift-invariant (Phased) GMM and the corresponding learning process is a Phased EM algorithm where each time series is re-phased in the **E** step. In experiments presented below we use Phased GMM directly in the feature space and generalize it so that each class has a separate covariance matrix.

We use the same experimental data to investigate the performance of the DP-GMT where the truncation level is set to be 30 and the concentration parameter α of the DP is set to be 1. The results are shown in Figure 7 and Table 2 where BIC-GMT means that the model order is chosen by BIC where the optimal k is chosen from 1 to 30. The poor performance of SCMT shows that a single center is not sufficient for this data. As evident from the graph the DP-GMT is not distinguishable from the BIC-GMT. The advantage of the DP model is that this equivalent performance is achieved with much reduced computational cost because the BIC procedure must learn many models and choose among them whereas the DP learns a single model.

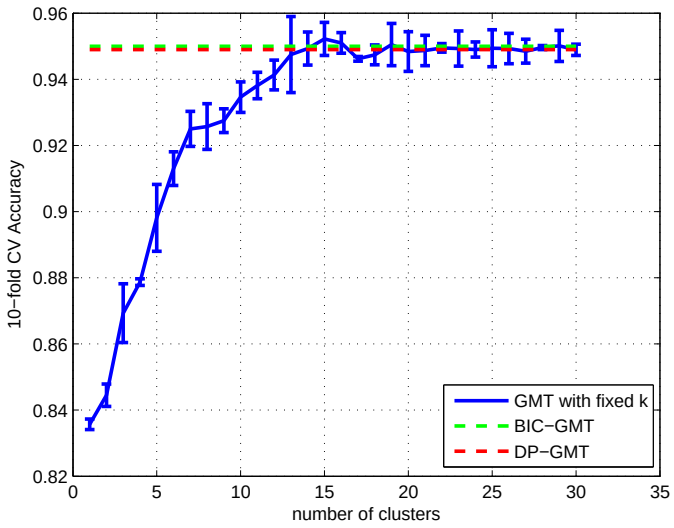

Fig. 7. OGLEII data: Comparison of model selection methods using dense sampled data. The plot shows the performance of GMT with varying k, BIC for the GMT model, and DP-GMT. For visual clarity we only include the standard deviations on the GMT plot.

	SCMT	GMT	DP-GMT	BIC-GMT
RESULTS	0.874 ± 0.008	0.952 ± 0.005	0.949 ± 0.005	0.950 ± 0.002

Table 2. Accuracies with standard deviations reported on OGLEII dataset.

Classification using sparse-sampled time series The OGLEII data set is in some sense a "nice" subset of the data from its corresponding star survey. Stars with small number of samples are often removed in pre-processing steps. For example, [34] developed full system to process the MACHO catalog and applied the kernel method to classify stars. In its pipeline, part of the preprocessing rejected 3.6 million light curves of the approximate 25 million because of insufficient number of observations. The proposed method potentially provides a way to include these instances in the classification process. In the second experiment, we demonstrate the performance of the proposed method on times series with sparse samples. Similar to the synthetic data, we started from sub-sampled versions of the original time series to simulate the condition that we would encounter in further star surveys.[7] As in the previous experiment, each time series is universally phased, normalized and linearly-interpolated to length 50 to be plugged into GMM and 1-NN as well as the phased GMM mentioned above. The RBF kernel is used for the proposed method and we use model order 15 as above. Moreover, the performance for PKmeans is also presented, where the classification step is as follows: we learn the PKmeans model with $k = 15$ for each class and then the label of a new example is assigned to be the same as its closest centroid's label. PKmeans is also restarted 5 times and the best clustering is used for classification.

The results are shown in Figure 8. As can be easily observed, when each time series has sparse samples (i.e., number of samples per task is less than 30), the proposed method has a significant advantage over the other methods. As the number of samples per task increases, the proposed method improves fast and performs close to its optimal performance given by previous experiment. Three additional aspects that call for discussion can be seen in the figure. First, note that for all three methods, the performance with dense data is lower than results in Table 1. This can be explained by fact that the data set obtained by the interpolation of the sub-sampled measurements contains less information than that interpolated from the original measurements. Second, notice that the Phased EM algorithm always outperforms the GMM plus UP demonstrating that re-phasing the time series inside the EM algorithm improves the results. Third, when the number of samples increases, the performance of the Phased EM gradually catches up and becomes better than the proposed method when each task has more than 50 samples. GMM plus universal phasing (UP) also achieves better performance when time series are densely sampled. One reason for the

[7] For the proposed method, we clip the samples to a fine grid of 200 equally spaced time points on $[0, 1]$, which is also the set of allowed time shifts. This avoids having a very high dimensional $\breve{\boldsymbol{x}}$, e.g. over 18000 for OGLEII, which is not feasible for any kernel based regression method that relies on solving linear systems.

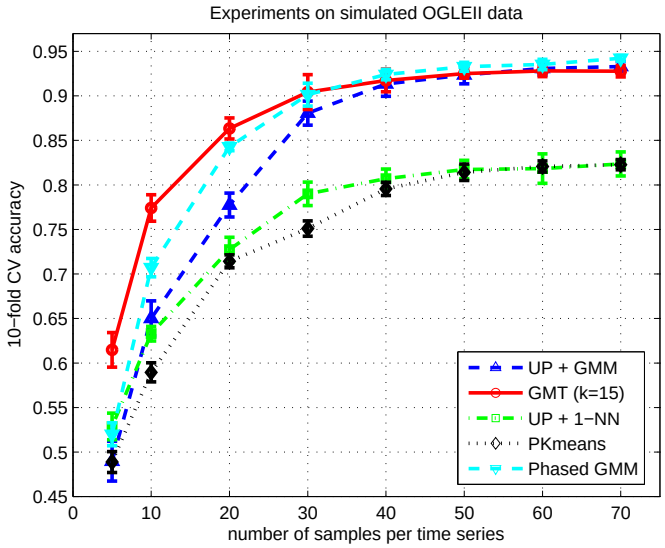

Fig. 8. OGLEII data: Comparison of algorithms with sparsely sampled data

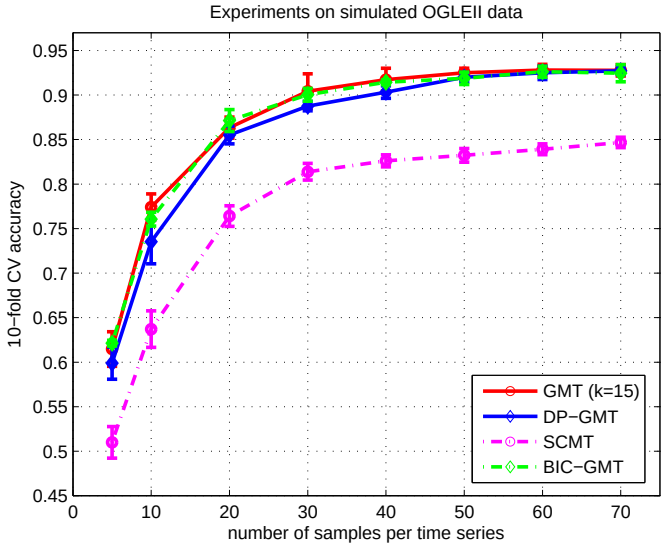

Fig. 9. OGLEII data: Comparison of algorithms with sparsely sampled data

performance difference is the difference in the way the kernel is estimated. In Figure 8 GMT uses the parametric form of the kernel which is less expressive than getting precise estimates for every $\mathcal{K}(t_1, t_2)$. The GMM uses the non-parametric form which, given sufficient data, can lead to better estimates. A second reason can be attributed to the sharing of the covariance function in our model where the GMM and the Phased GMM do not apply this constraint.

Finally, we use the same experimental setting to compare the performance of various mode selection models. The results are shown in Figure 9. The performance of BIC is not distinguishable from the optimal k selected in hindsight. The performance of DP is slightly lower but it comes close to these models.

To summarize, we conclude from the experiments with astronomy data that Phased EM is appropriate with densely sampled data but that the GMT and its variants should be used when data is sparsely and non-synchronously sampled. In addition BIC coupled with GMT performs excellent model selection and DP does almost as well with a much reduced computational complexity.

Class discovery: We show the potential of our model for class discovery by running two version of the GMT model on the joint data set of the three classes (not using the labels). Then, each cluster is labeled according to the majority class of the instances that belong to the center. For a new test point, we determine which cluster it belongs to via the MAP probability and its label is given by the cluster that it is assigned to. We run 10 trials with different random initializations. In accordance with previous experiments that used 15 components per class we run GMT with model order of 45. We also run DP-GMT with a truncation level set to 90. The GMT obtains accuracy and standard deviation of $[0.895, 0.010]$ and the DP models obtains accuracy and standard deviation of $[0.925, 0.013]$. Note that it is hard to compare between the results because of the different model orders used. Rather than focus on the difference, the striking point is that we obtain almost pure clusters without using any label information. Given the size of the data set and the relatively small number of clusters this is a significant indication of the potential for class discovery in astrophysics.

7 Related Work

Classification of time series has attracted an increasing amount of interest in recent years due to its wide range of potential applications, for example ECG diagnosis [35], EEG diagnosis [16], and Speech Recognition [36]. Common methods choose some feature based representation or distance function for the time series (for example the sampled time points, or Fourier or wavelet coefficients as features and dynamic time warping for distance function) and then apply some existing classification method [37, 38]. Our approach falls into another category, that is, model-based classification where the time series are assumed to be generated by a probabilistic model and examples are classified using maximum likelihood or MAP estimates. A family of such models, closely related to the GMT, is discussed in detail below. Another common approach uses Hidden

Markov models as a probabilistic model for sequence classification, and this has been applied to time series as well [39].

Learning Gaussian processes from multiple tasks has previously been investigated in the the hierarchical Bayesian framework, where a group of related tasks are assumed to share the same prior. Under this assumption, training points across all tasks are utilized to learn a better covariance function via the EM algorithm [7, 8]. In addition, [16] extended the work of [8] to a non-parametric mixed-effect model where each task can have its own random effect. Our model is based on the same algorithmic approach where the values of the function for each task at its corresponding points (i.e. $\{\mathbf{f}_j\}$ in our model) are considered as hidden variables. Furthermore, the proposed model is a natural generalization of [8] where the fixed-effect function is sampled from a mixture of regression functions each of which is a realization of a common Gaussian process. Along a different dimension, our model differs from the infinite mixtures of Gaussian processes model for clustering [40] in two aspects: first, instead of using zero mean Gaussian process, we allow the mean functions to be sampled from another Gaussian process; second, the individual variation in our model serves as the covariance function in their model but all mixture components share the same kernel.

Although having a similar name, the Gaussian process *mixture of experts* model focuses mainly on the issues of non-stationarity in regression [41, 42]. By dividing the input space into several (even infinite) regions via a gating network, the Gaussian process mixture of expert model allows different Gaussian processes to make predictions for different regions.

In terms of the clustering aspect, our work is most closely related to the so-called *mixture of regressions* [43, 11, 44, 45]. The name comes from the fact that these approaches substitute component density models with conditional regression density models in the framework of standard mixture model. For phased time series, [45] first proposed the regression-based mixture model where they used Polynomial and Kernel regression models for the mean curves. Further, [11] integrated the linear random effects models with mixtures of regression functions. In their model, each time series is sampled by a parametric regression model whose parameters are generated from a Gaussian distribution. To incorporate the time shifts, [46] proposed a shift-invariant Gaussian mixture model for multidimensional time series. They constrained the covariance matrices to be diagonal to handle the non-synchronous case. They also treated time shifts as hidden variables and derived the EM algorithm under full Bayesian settings, i.e. where each parameter has a prior distribution. Furthermore, [43] developed a generative model for misaligned curves in a more general setting. Their joint clustering-alignment model also assumes a normal parametric regression model for the cluster labels, and Gaussian priors on the hidden transformation variables which consist of shifting and scaling in both the time and magnitude. Our model extends the work of [11] to admit non-parametric Bayesian regression mixture models and at the same time handle the non-phased time series. If the group effects are assumed to have a flat prior, our model differs from [46] in the

following two aspects in addition to the difference of Bayesian treatment. First, our model does not include the time shifts as hidden variables but instead estimates them as parameters. Second, we can handle shared full covariance matrix instead of diagonal ones by using a parametric form of the kernel. On the other hand, given the time grid $\breve{\boldsymbol{x}}$, we can design the kernel for individual variations as $\mathcal{K}(\breve{x}_i, \breve{x}_j) = a_i \delta_{ij}(\breve{x}_i, \breve{x}_j), i, j = 1, \cdots, \mathbb{N}$. Using this choice, our model is the same as [46] with shared diagonal covariance matrix. In summary, our model allows a more flexible structure of the covariance matrix that can treat synchronized and non-synchronized time series in a unified framework, but at the same time it is constrained to have the same covariance matrix across all clusters.

8 Conclusion

We developed a novel Bayesian nonparametric multi-task learning model (GMT) where each task is modeled as a sum of a group-specific function and an individual task function with a Gaussian process prior. We also extended the model such that the number of groups is not bounded using a Dirichlet process mixture model (DP-GMT). We derive efficient EM and variational EM algorithms to learn the parameters of the models and demonstrated their effectiveness using experiments in regression, classification and class discovery. Our models are particularly useful for sparsely and non-synchronously sampled time series data, and model selection can be effectively performed with these models.

There are several natural directions for future work. For application in the astronomy context it is important to consider all steps of processing and classification of a new sky survey so as to provide an end to end system. Therefore, two important issues to be addressed in future work include incorporating the period estimation phase into the method and developing an appropriate method for abstention in the classification step. It would also be interesting to develop a corresponding discriminative model extending [5] to the GP context. Finally, one of the drawbacks of the GP based methods is the computational complexity which is too high for large scale problems. For example, in the experiments on sparse OGLEII data, we had to resample the data on a fine grid to avoid performing Cholesky decomposition for high dimensional matrices. Therefore, an important direction for future work is to find non-trivial sparse GP approximations that yield good performance with the GMT model.

Acknowledgments

This research was partly supported by NSF grant IIS-0803409. The experiments in this paper were performed on the Odyssey cluster supported by the FAS Research Computing Group at Harvard and the Tufts Linux Research Cluster supported by Tufts UIT Research Computing.

References

1. Bi, J., Xiong, T., Yu, S., Dundar, M., Rao, R.: An improved multi-task learning approach with applications in medical diagnosis. European Conference on Machine Learning and Knowledge Discovery in Databases (2008) 117–132
2. Dinuzzo, F., Pillonetto, G., De Nicolao, G.: Client-server multi-task learning from distributed datasets. Arxiv preprint arXiv:0812.4235 (2008)
3. Bickel, S., Bogojeska, J., Lengauer, T., Scheffer, T.: Multi-task learning for HIV therapy screening. In: Proceedings of the 25th international conference on Machine learning, ACM (2008) 56–63
4. Evgeniou, T., Micchelli, C., Pontil, M.: Learning multiple tasks with kernel methods. Journal of Machine Learning Research **6**(1) (2006) 615–637
5. Xue, Y., Liao, X., Carin, L., Krishnapuram, B.: Multi-task learning for classification with Dirichlet process priors. Journal of Machine Learning Research **8** (2007) 35–63
6. Gelman, A.: Bayesian data analysis. CRC press (2004)
7. Yu, K., Tresp, V., Schwaighofer, A.: Learning Gaussian processes from multiple tasks. In: Proceedings of the 22nd international conference on Machine learning, ACM (2005) 1012–1019
8. Schwaighofer, A., Tresp, V., Yu, K.: Learning Gaussian process kernels via hierarchical Bayes. Advances in Neural Information Processing Systems **17** (2005) 1209–1216
9. Pillonetto, G., Dinuzzo, F., De Nicolao, G.: Bayesian Online Multitask Learning of Gaussian Processes. IEEE Transactions on Pattern Analysis and Machine Intelligence **32**(2) (2010) 193–205
10. Rebbapragada, U., Protopapas, P., Brodley, C.E., Alcock, C.: Finding anomalous periodic time series. Machine Learning **74**(3) (2009) 281–313
11. Gaffney, S.J., Smyth, P.: Curve clustering with random effects regression mixtures. In: Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics. (2003)
12. Evgeniou, T., Pontil, M., Poggio, T.: Regularization networks and support vector machines. Advances in Computational Mathematics **13**(1) (2000) 1–50
13. Scholkopf, B., Smola, A.J.: Learning with kernels. MIT Press (2002)
14. Micchelli, C.A., Pontil, M.: On learning vector-valued functions. Neural Computation **17**(1) (2005) 177–204
15. Rasmussen, C.E.: Gaussian processes in machine learning. Advanced Lectures on Machine Learning (2006) 63–71
16. Lu, Z., Leen, T., Huang, Y., Erdogmus, D.: A reproducing kernel Hilbert space framework for pairwise time series distances. In: Proceedings of the 25th international conference on Machine learning, ACM New York, NY, USA (2008) 624–631
17. Seeger, M.: Gaussian processes for machine learning. International Journal of Neural Systems **14**(2) (2004) 69–106
18. Teh, Y.W.: Dirichlet processes. In: Encyclopedia of Machine Learning. Springer (2010)
19. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum-likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society **B39** (1977) 1–38
20. Stein, E., Shakarchi, R.: Real analysis: measure theory, integration, and Hilbert spaces. Princeton University Press (2005)

21. Protopapas, P., Giammarco, J.M., Faccioli, L., Struble, M.F., Dave, R., Alcock, C.: Finding outlier light curves in catalogues of periodic variable stars. Monthly Notices of the Royal Astronomical Society **369** (2006) 677–696
22. Sethuraman, J.: A constructive definition of Dirichlet priors. Statistica Sinica **4**(2) (1994) 639–650
23. Ishwaran, H., James, L.: Gibbs sampling methods for stick-breaking priors. Journal of the American Statistical Association **96**(453) (2001) 161–173
24. Wainwright, M., Jordan, M.: Graphical models, exponential families, and variational inference. Foundations and Trends® in Machine Learning **1**(1-2) (2008) 1–305
25. Blei, D., Jordan, M.: Variational inference for Dirichlet process mixtures. Bayesian Analysis **1**(1) (2006) 121–144
26. Rasmussen, C., Nickisch, H.: Gaussian Processes for Machine Learning (GPML) Toolbox. Journal of Machine Learning Research **11** (2010) 3011–3015
27. Udalski, A., Szymanski, M., Kubiak, M., Pietrzynski, G., Wozniak, P., Zebrun, Z.: Optical gravitational lensing experiment. photometry of the macho-smc-1 microlensing candidate. Acta Astronomica **47** (1997) 431–436
28. Alcock, C., et al.: The MACHO Project - a Search for the Dark Matter in the Milky-Way. In Soifer, B.T., ed.: Sky Surveys. Protostars to Protogalaxies. Volume 43 of Astronomical Society of the Pacific Conference Series. (1993) 291–296
29. Faccioli, L., Alcock, C., Cook, K., Prochter, G.E., Protopapas, P., Syphers, D.: Eclipsing Binary Stars in the Large and Small Magellanic Clouds from the MACHO Project: The Sample. Astronomy Journal **134** (2007) 1963–1993
30. Hodapp, K.W., et al.: Design of the Pan-STARRS telescopes. Astronomische Nachrichten **325** (2004) 636–642
31. Starr, B.M., et al.: LSST Instrument Concept. In Tyson, J.A., Wolff, S., eds.: Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series. Volume 4836 of Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series. (December 2002) 228–239
32. Soszynski, I., Udalski, A., Szymanski, M.: The Optical Gravitational Lensing Experiment. Catalog of RR Lyr Stars in the Large Magellanic Cloud 06. Acta Astronomica **53** (2003) 93–116
33. Wachman, G., Khardon, R., Protopapas, P., Alcock, C.: Kernels for Periodic Time Series Arising in Astronomy. In: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II, Springer (2009) 489–505
34. Wachman, G.: Kernel Methods and Their Application to Structured Data. PhD thesis, Tufts University (2009)
35. Wei, L., Keogh, E.: Semi-supervised time series classification. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM New York, NY, USA (2006) 748–753
36. Povinelli, R.J., Johnson, M.T., Lindgren, A.C., Ye, J.: Time series classification using Gaussian mixture models of reconstructed phase spaces. IEEE Transactions on Knowledge and Data Engineering **16**(6) (2004) 779–783
37. Osowski, S., Hoai, L., Markiewicz, T.: Support vector machine-based expert system for reliable heartbeat recognition. IEEE Transactions on Biomedical Engineering **51**(4) (2004) 582–589
38. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and mining of time series data: experimental comparison of representations and distance measures. Proceedings of the VLDB Endowment archive **1**(2) (2008) 1542–1552

39. Kim, S., Smyth, P.: Segmental hidden Markov models with random effects for waveform modeling. Journal of Machine Learning Research **7** (2006) 969
40. Jackson, E., Davy, M., Doucet, A., Fitzgerald, W.: Bayesian unsupervised signal classification by dirichlet process mixtures of gaussian processes. In: IEEE International Conference on Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. Volume 3. (2007)
41. Rasmussen, C.E., Ghahramani, Z.: Infinite mixtures of Gaussian process experts. In: Advances in neural information processing systems 14: proceedings of the 2001 conference, MIT Press (2002) 881–888
42. Tresp, V.: Mixtures of Gaussian processes. Advances in Neural Information Processing Systems (2001) 654–660
43. Gaffney, S.J., Smyth, P.: Joint probabilistic curve clustering and alignment. Advances in neural information processing systems **17** (2005) 473–480
44. Gaffney, S.J.: Probabilistic curve-aligned clustering and prediction with regression mixture models. PhD thesis, University of California, Irvine (2004)
45. Gaffney, S.J., Smyth, P.: Trajectory clustering with mixtures of regression models. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (1999) 63–72
46. Chudova, D., Gaffney, S.J., Mjolsness, E., Smyth, P.: Translation-invariant mixture models for curve clustering. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2003) 79–88